

微信小程序

开发必备100Tip

刘刚 编著

電子工業出版社
Publishing House of Electronics Industry
北京•BEIJING

VISIT…

LANZAROTE Caliente.COM

内 容 简 介

本书提供了微信小程序开发必备的 100 个 Tip 和一个完整的电商案例京东购物微信小程序，100 个 Tip 分为四类：关于微信小程序的框架及工具 16 个 Tip、关于微信小程序丰富的组件 24 个 Tip、关于微信小程序必备 API 技能 50 个 Tip、关于微信小程序高级交互设计 10 个 Tip，每一个 Tip 都会有详细的介绍和相关的示例，方便开发人员快速学习这些 Tip，同时也可以作为工具书指南进行查询，应用到哪个知识点可以到这里快速的查询进行使用。京东购物微信小程序是综合应用这些 Tip，将这些 Tip 综合应用到实际项目开发过程中。通过学习 100 个 Tip 和项目案例，让读者全面、深入、透彻地理解微信小程序开发，提高微信小程序开发能力和项目实战能力。

本书内容丰富，实例典型，实用性强，适合各个层次想要学习微信小程序开发技术的人员阅读，本书也可以作为微信小程序开发查询工具书使用，方便快速查询微信小程序开发所应用到的知识点。

未经许可，不得以任何方式复制或抄袭本书之部分或全部内容。
版权所有，侵权必究。

图书在版编目（CIP）数据

微信小程序开发必备 100Tip / 刘刚编著. —北京：电子工业出版社，2017.10
ISBN 978-7-121-32691-2

Ⅰ. ①微… Ⅱ. ①刘… Ⅲ. ①移动终端－应用程序－程序设计 Ⅳ. ①TN929.53

中国版本图书馆 CIP 数据核字(2017)第 223271 号

责任编辑：安　娜
印　　刷：
装　　订：
出版发行：电子工业出版社
　　　　　北京市海淀区万寿路 173 信箱　邮编：100036
开　　本：787×980　1/16　印张：23.5　字数：438 千字
版　　次：2017 年 10 月第 1 版
印　　次：2017 年 10 月第 1 次印刷
定　　价：75.00 元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888，88258888。

质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

本书咨询联系方式：010-51260888-819，faq@phei.com.cn。

前　言

为什么要写这本书？

微信小程序从正式发布后越来越受很多软件公司和开发人员的使用，目前图书市场上关于微信小程序的书籍还很匮乏，真正能把微信小程序开发必备技能提炼出来的少之又少，本书提供了微信小程序开发必备的 100 个 Tip 和一个完整的电商案例——京东购物微信小程序。

100 个 Tip 分为四类：

- 关于微信小程序的框架及工具 16 个 Tip；
- 关于微信小程序丰富的组件 24 个 Tip
- 关于微信小程序必备 API 技能 50 个 Tip；
- 关于微信小程序高级交互设计 10 个 Tip。

每一个 Tip 都有详细的介绍和相关的示例，方便开发人员快速学习这些 Tip，同时也可以作为工具书指南进行查询，应用到哪个知识点可以到这里快速查询并使用。京东购物微信小程序是将这些 Tip 综合应用到实际项目开发过程中。通过学习 100 个 Tip 和项目案例，让读者全面、深入、透彻地理解微信小程序开发，提高微信小程序开发能力和项目实战能力。

本书有何特色？

1. 图文结合方式，逐步讲解

为了便于读者理解本书内容，提高学习效率，作者通过文字讲解及图表演示，逐步操作，避免枯燥乏味。

2. 提炼知识点、方便快速学习

本书针对微信小程序的框架和工具、组件、API、交互效果共提炼出 100 个常用开发

Tip，每个 Tip 都有详细的示例或者实际应用的场景，方便读者快速学习和查阅使用。

3. 由浅入深、理论与实战相结合

本书先讲解 100 个常用开发 Tip，从框架和工具的使用到组件的使用，然后是常用 API，再后是高级交互效果，最后使用一个完整案例贯穿知识点，由浅入深，一步步掌握微信小程序的开发。

4. 实战驱动，应用性强

本书每个开发 Tip 都有详细的示例或者实际应用场景，读者可根据示例快速进行知识点的学习和使用，通过知识点在实际场景的应用，指引读者在项目实战过程中如何应用这些 Tip。

5. 项目案例典型，实战性强，有较高的应用价值

本书提供了有代表性的高级交互效果和完整的典型案例。读者在实际项目开发过程中，可参考这些案例，缩短开发时间。

6. 提供完善的技术支持和售后服务

本书提供了专门的技术支持邮箱：yuan_kevin_525@126.com，读者在阅读本书过程中如有任何疑问都可以通过该邮箱获得帮助。

本书内容及知识体系

第 1 章　微信小程序的框架及工具

本章介绍了微信小程序的框架及工具使用 Tip 15 个，详细讲解了微信小程序框架和工具的使用，是小程序入门必备的知识点。

第 2 章　微信小程序的丰富的组件

本章介绍了微信小程序组件的使用 Tip 26 个，详细介绍了每个组件的使用以及实际项目应用场景，是小程序界面开发必备的知识点。

第 3 章　微信小程序必备 API 技能

本章介绍了微信小程序 API 使用 Tip 49 个，详细介绍微信小程序常用 API 的使用，是小程序进行界面与后台逻辑处理开发必备的知识点。

第 4 章　微信小程序高级交互设计

本章介绍了微信小程序高级交互设计 Tip 10 个，介绍了 10 类高级交互效果，是小程序交互设计必备的知识点。

第 5 章　京东购物微信小程序案例

本章详细讲解了微信小程序的一个典型案例，通过对案例的学习，读者可掌握各个知识点的应用以及实际项目的开发过程和设计思路。

适合阅读本书的读者

- 微信开发者
- 前端开发工程师
- 学生
- 网站站长、待业青年

阅读本书的建议

- 本书可以作为工具书使用，查询各个知识点、交互效果的使用。
- 没有基础的读者，可以按照章节顺序进行学习，由浅入深，逐个突破。
- 有一定基础的读者，可以根据实际情况有重点地选择阅读各个模块和项目案例。
- 对于每一个模块和项目案例，先自己思考一下实现的思路，然后再阅读，学习效果会更好。

轻松注册成为博文视点社区用户（www.broadview.com.cn），扫码直达本书页面。

- **提交勘误：**您对书中内容的修改意见可在 提交勘误 处提交，若被采纳，将获赠博文视点社区积分（在您购买电子书时，积分可用来抵扣相应金额）。
- **交流互动：**在页面下方 读者评论 处留下您的疑问或观点，与我们和其他读者一同学习交流。

页面入口：http://www.broadview.com.cn/32691



目　　录

第 1 章　微信小程序的框架及工具 1
1.1　app.json 配置 1
1.1.1　决定页面文件路径 2
1.1.2　配置窗口表现 3
1.1.3　配置 tab 标签导航 5
1.1.4　设置网络超时时间 8
1.1.5　配置 debug 模式 9
1.2　App()函数使用 10
1.3　Page()函数使用 12
1.3.1　初始化数据 13
1.3.2　生命周期函数 13
1.3.3　相关事件函数 14
1.4　数据绑定 14
1.5　wx:if 条件判断 15
1.6　wx:for 列表渲染 16
1.7　定义和使用模板 17
1.8　import 引用模板 18
1.9　include 引用文件 19
1.10　Console 面板 20
1.11　Sources 面板 21
1.12　Network 面板 22
1.13　Storage 面板 24
1.14　Appdata 面板 24

1.15 Wxml 面板 26
1.16 Sensor 面板 27

第 2 章　微信小程序丰富的组件 28

2.1 view 视图容器组件......28
2.2 scroll-view 可滚动视图容器组件 31
2.3 swiper 滑块视图容器组件......34
2.3.1 海报轮播效果 35
2.3.2 页签切换效果 36
2.4 movable-view 可移动的视图容器组件......40
2.5 icon 图标组件 42
2.6 text 文本组件 44
2.7 progress 进度条组件......45
2.8 button 按钮组件 46
2.9 checkbox 多项选择器组件 48
2.10 radio 单项选择器组件 57
2.11 input 单行输入框组件 59
2.12 textarea 多行输入框组件......61
2.13 label 改进表单组件......63
2.14 picker 选择器组件 70
2.14.1 picker 普通选择器组件 71
2.14.2 picker 时间选择器组件 73
2.14.3 picker 日期选择器组件 74
2.15 picker-view 嵌入页面的滚动选择器 76
2.16 slider 滑动选择器组件......78
2.17 switch 开关选择器组件......80
2.18 form 表单组件......85
2.19 navigator 页面链接导航组件 92
2.20 audio 音频组件 95
2.21 image 图片组件......102
2.22 video 视频组件 104
2.23 map 地图组件 106
2.24 canvas 画布组件 112

第 3 章 微信小程序必备 API 技能 118
3.1 wx.request 发起网络请求 118
3.2 wx.uploadFile 上传文件 122
3.3 wx.downloadFile 下载文件......123
3.4 WebSocket 会话连接......125
3.5 wx.chooseImage 选择图片 133
3.6 wx.previewImage 预览图片......134
3.7 wx.getImageInfo 获取图片信息 135
3.8 wx.saveImageToPhotosAlbum 保存图片到相册 136
3.9 wx.startRecord 开始录音和 wx.stopRecord 结束录音 138
3.10 voice 音频播放控制......139
3.11 wx.createAudioContext 音频组件控制 140
3.12 wx.playBackgroundAudio 播放音乐 141
3.13 wx.seekBackgroundAudio(OBJECT)控制音乐播放进度 143
3.14 wx.getBackgroundAudioPlayerState 获取音乐播放状态 143
3.15 wx.createVideoContext(videoId)视频组件控制......145
3.16 wx.chooseVideo(OBJECT)选择视频 146
3.17 wx.saveVideoToPhotosAlbum 保存视频到相册 147
3.18 wx.saveFile 保存文件 148
3.19 wx.getSavedFileList 获取本地文件列表......149
3.20 wx. getSavedFileInfo 获取本地文件信息 151
3.21 wx. removeSavedFile 删除本地文件......153
3.22 wx. openDocument 打开文档 154
3.23 wx.setStorage 和 wx.setStorageSync 本地数据缓存......155
3.24 wx.getStorage 和 wx.getStorageSync 获取本地数据缓存......157
3.25 wx.getStorageInfo 和 wx.getStorageInfoSync 获取缓存信息 159
3.26 清理本地缓存数据 161
3.27 wx.getLocation 获取位置 162
3.28 wx. chooseLocation 选择位置 167
3.29 wx.openLocation 打开内置地图......168
3.30 获取设备系统信息 169
3.31 获取网络状态 172
3.32 wx.onAccelerometerChange 加速度计 172

3.33　wx.onCompassChange 监听罗盘数据 173
3.34　wx.makePhoneCall 拨打电话和 wx.scanCode 扫码 173
3.35　剪贴板和屏幕亮度 175
3.36　wx.addPhoneContact 添加联系人 176
3.37　wx.showToast 和 wx.showLoading 显示提示框 178
3.38　wx.showModal 显示模态弹窗 180
3.39　wx.showActionSheet 显示操作菜单 182
3.40　设置导航条 183
3.41　wx.createAnimation 创建动画 184
3.42　获取第三方平台数据 190
3.43　wx.login 登录 191
3.44　wx.authorize 授权 193
3.45　wx.getUserInfo 用户信息 195
3.46　wwx.requestPayment 微信支付 197
3.47　模板消息 199
3.48　转发信息 202
3.49　wx.chooseAddress 收货地址 203
3.50　卡券 204

第 4 章　微信小程序高级交互设计 207

4.1　猫眼电影：海报轮播效果设计 207
4.2　许鲜全国送：页签切换效果设计 210
4.3　新闻频道：水平滑动效果设计 215
4.4　京东到家：手风琴导航切换效果设计 219
4.5　链家：列表式导航设计 223
4.6　支付宝：九宫格导航设计 230
4.7　58 同城：区域条件检索设计 237
4.8　12306 火车票：底部固定效果设计 242
4.9　当当购物：图文列表设计 245
4.10　中信信用卡申请：向导型表单设计 255

第 5 章　综合案例：京东购物小程序 278

5.1　需求描述 279

5.2　相关知识点 285
5.3　商品数据准备 286
5.4　首页设计 290
5.5　商品详情页设计 305
5.6　立即购买页面 320
5.7　商品搜索设计 329
5.8　显示购物车商品设计 339
5.9　我的订单页面设计 350
5.10　优惠券设计 356
5.11　小结 366

第 1 章　微信小程序的框架及工具

微信小程序的框架及工具的使用，是进行小程序开发的第一步，首先要理解小程序的框架配置，包括 app.json 配置、App()函数的使用、Page()函数的使用、数据如何绑定到页面上、条件判断和列表渲染、模板的使用、import 引用模板，以及 include 引用文件，同时还要学会工具的使用，包括 Console 面板、Sources 面板、Network 面板、Storage 面板、Appdata 面板、Wxml 面板以及 Sensor 面板的使用。

1.1　app.json 配置

app.json 配置是进行微信小程序开发必会的一个功能配置，对于微信小程序而言，它是一个全局的配置，有五个主要功能：决定页面文件路径，配置窗口的表现，配置 tab 标签导航，设置网络超时时间以及配置 debug 模式。配置代码如下所示。

```
{
   "pages":[
      "pages/index/index",
      "pages/me/me"
   ],
   "window":{
      "backgroundTextStyle":"light",
      "navigationBarBackgroundColor": "#D53C3E",
      "navigationBarTitleText": "窗口表现",
      "navigationBarTextStyle":"white",
      "enablePullDownRefresh":"true",
      "backgroundColor":"black"
   },
   "tabBar": {
      "list": [{
         "pagePath": "pages/index/index",
         "text": "首页",
         "iconPath": "pages/images/bar/index-0.jpg",
```

```
            "selectedIconPath": "pages/images/bar/index-1.jpg"
        },{
            "pagePath": "pages/me/me",
            "text": "我的",
            "iconPath": "pages/images/bar/me-0.jpg",
            "selectedIconPath": "pages/images/bar/me-1.jpg"
        }]
    },
    "networkTimeout": {
        "request": 20000,
        "connectSocket": 20000,
        "uploadFile": 20000,
        "downloadFile": 20000
    },
    "debug": true
}
```

1.1.1 决定页面文件路径

微信小程序的页面路径需要在 app.json 配置文件中配置一下，属性名为“pages”，以数组的形式存放页面文件路径。文件路径以文件名进行命名，数组的第一项代表小程序的初始页面。当小程序中新增/减少页面时，都需要对 pages 数组进行修改。文件名不需要写文件后缀，因为框架会自动去寻找路径，对`.json`、`.js`、`.wxml`和`.wxss`四个文件进行整合。具体代码如下所示。

```
"pages":[
        "pages/index/index",
        "pages/me/me"
    ]
```

在 app.json 配置完页面路径后，会生成文件路径对应名称的文件夹和 4 个对应名称的文件（.js 文件、.json 文件、.wxml 文件、.wxss 文件），如图 1.1 所示。

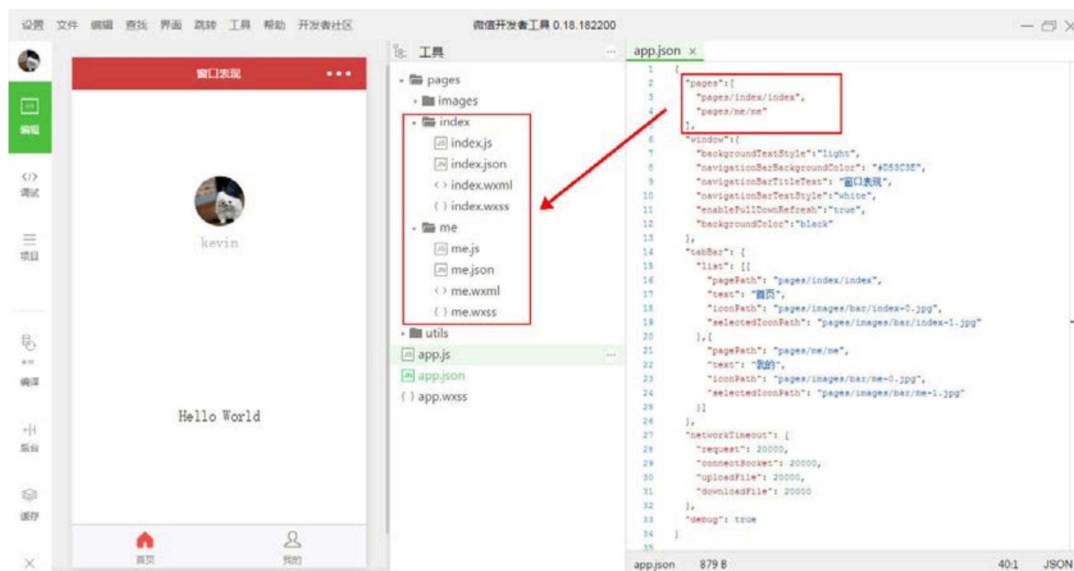


图 1.1　页面路径

注意：页面路径以文件名称来创建，“pages/index/index”这样的页面路径是推荐的，而“pages/common/index”这样的页面路径是不推荐的，因为不便于微信小程序创建文件夹和文件，也不方便我们维护。

1.1.2　配置窗口表现

app.json 文件可以配置窗口的表现，包括导航栏背景色、导航栏标题颜色、导航栏标题文字内容、窗口的背景色、下拉背景字体、loading 图的样式，以及是否开启下拉刷新设置。这些功能的配置都放置在 window 这个对象里，具体代码如下所示。

```
"window":{
    "backgroundTextStyle":"light",
    "navigationBarBackgroundColor": "#D53C3E",
    "navigationBarTitleText": "窗口表现",
    "navigationBarTextStyle":"white",
    "enablePullDownRefresh":"true",
    "backgroundColor":"black"
 }
```

窗口属性说明如下。

（1）navigationBarBackgroundColor：导航栏背景颜色，如#00000。

（2）navigationBarTextStyle：导航栏标题颜色，仅支持 black/white。

（3）navigationBarTitleText：导航栏标题文字内容。

（4）backgroundColor：窗口的背景色。

（5）backgroundTextStyle：下拉背景字体、loading 图的样式，仅支持 dark/light。

（6）enablePullDownRefresh：是否开启下拉刷新。

界面效果如图 1.2 所示。

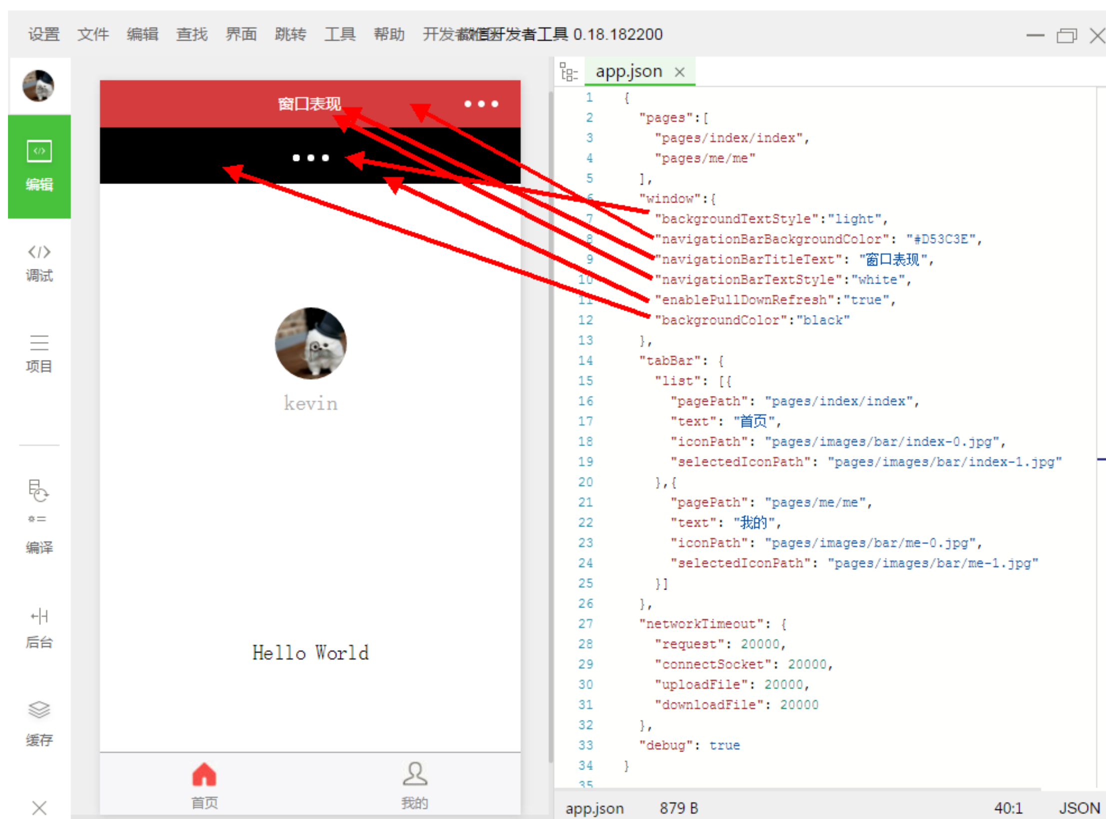


图 1.2　界面效果

1.1.3　配置 tab 标签导航

手机上安装的很多 App 软件，比较流行的一种导航方式就是设置底部标签导航，一般底部标签导航有 2~5 个，微信小程序也支持这样的导航方式，通过配置 tab 标签导航来支持这种流行的导航方式，像智行 12306 抢票软件，底部标签有四个，分别是火车票、飞机票、汽车票和个人中心，如图 1.3 所示。



图 1.3　智行 12306 抢票软件

微信小程序配置 tab 标签导航，可以配置底部标签导航和顶部标签导航，通过 tabBar 来配置。tabBar 是一个数组，只能配置最少 2 个、最多 5 个 tab，tab 按数组的顺序排序。当 position 属性值设置为 top 时，为顶部标签导航；当 position 属性值设置为 bottom 或者不设置时，为底部标签导航。

tabBar 的属性如下所示。

（1）color：tab 上的文字默认颜色。

（2）selectedColor：tab 上的文字选中时的颜色。

（3）backgroundColor：tab 的背景色。

（4）borderStyle：tabbar 上边框的颜色，仅支持 black/white。

（5）position：可选值 bottom、top。

（6）list：tab 的列表，最少 2 个、最多 5 个 tab，数组中的每一项都是一个对象。

（7）pagePath：页面路径，必须在 pages 中先定义。

（8）text：tab 上按钮文字。

（9）iconPath：图片路径，icon 大小限制为 40kb，建议尺寸为 81px × 81px。当 postion 为 top 时，此参数无效。

（10）selectedIconPath：选中时的图片路径，icon 大小限制为 40kb，建议尺寸为 81px × 81px。当 postion 为 top 时，此参数无效。

1．配置底部标签导航

具体代码如下所示。

```
"tabBar": {
      "color":"#000000",
      "selectedColor": "#D53C3E",
      "backgroundColor": "#ffffff",
      "borderStyle": "black",
      "position":"bottom",
      "list": [{
         "pagePath": "pages/index/index",
         "text": "首页",
         "iconPath": "pages/images/bar/index-0.jpg",
         "selectedIconPath": "pages/images/bar/index-1.jpg"
      },{
         "pagePath": "pages/me/me",
         "text": "我的",
         "iconPath": "pages/images/bar/me-0.jpg",
         "selectedIconPath": "pages/images/bar/me-1.jpg"
      }]
   }
```

界面效果如图 1.4 所示。

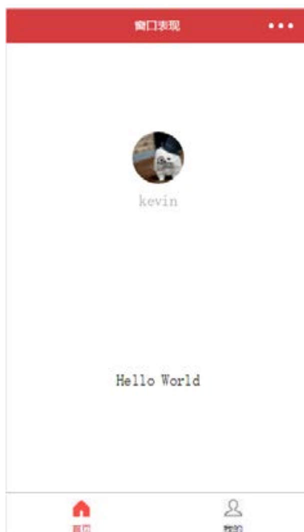


图 1.4　底部标签导航

2．配置顶部标签导航

具体代码如下所示。

```
"tabBar": {
      "color":"#000000",
      "selectedColor": "#D53C3E",
      "backgroundColor": "#ffffff",
      "borderStyle": "black",
      "position":"top",
      "list": [{
         "pagePath": "pages/index/index",
         "text": "首页",
         "iconPath": "pages/images/bar/index-0.jpg",
         "selectedIconPath": "pages/images/bar/index-1.jpg"
      },{
         "pagePath": "pages/me/me",
         "text": "我的",
         "iconPath": "pages/images/bar/me-0.jpg",
         "selectedIconPath": "pages/images/bar/me-1.jpg"
      }]
   }
```

界面效果如图 1.5 所示。

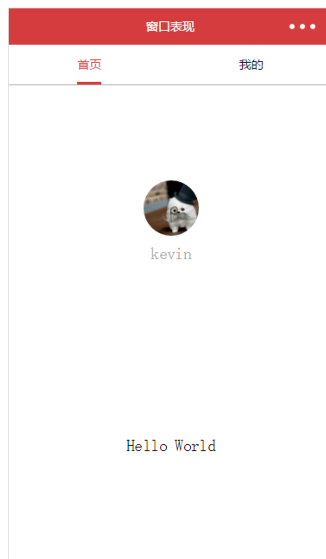


图 1.5 顶部标签导航

1.1.4 设置网络超时时间

微信小程序设置网络超时时间是通过 networkTimeout 对象来设置的，可以设置网络请求的超时时间、WebSocket 的超时时间、uploadFile 文件上传超时时间和 downloadFile 文件下载超时时间，它的属性如下所示。

（1）request：wx.request 的超时时间，单位毫秒，默认为 60000。

（2）connectSocket：wx.connectSocket 的超时时间，单位毫秒，默认为 60000。

（3）uploadFile：wx.uploadFile 的超时时间，单位毫秒，默认为 60000。

（4）downloadFile：wx.downloadFile 的超时时间，单位毫秒，默认为 6000。

配置网络超时时间，具体代码如下所示。

```
"networkTimeout": {
    "request": 20000,
    "connectSocket": 20000,
    "uploadFile": 20000,
    "downloadFile": 20000
  }
```

1.1.5　配置 debug 模式

配置 debug 模式是在开发者工具的控制台面板，调试信息并以 info 的形式给出，其信息有 Page 的注册、页面路由、数据更新和事件触发 。可以帮助开发者快速定位一些常见的问题。具体代码如下所示：

```
"debug": true
```

（1）开启 debug 模式，控制台面板的调试信息如图 1.6 所示。

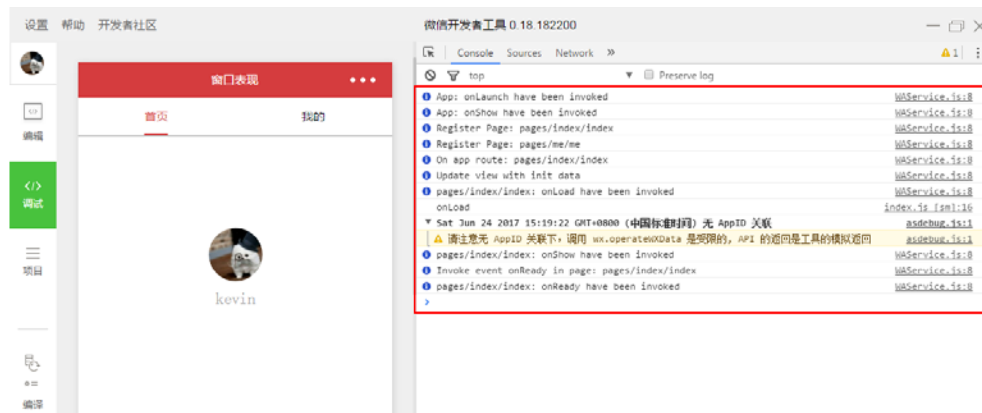


图 1.6　开启 debug 模式

（2）没有开启 debug 模式，控制台面板的调试信息如图 1.7 所示。

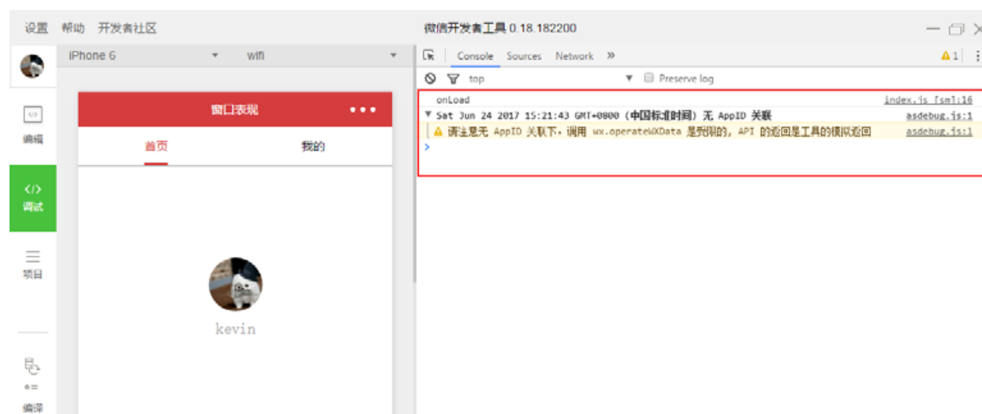


图 1.7　没有开启 debug 模式

从图 1.6 和图 1.7 中可以看出，开启 debug 模式时，调试信息会更详细、更全面，方便

在程序开发过程中进行调试。如果微信小程序正式运行，则需要把 debug 模式关闭。

1.2　App()函数使用

App()函数用来注册一个微信小程序，它被放置在 app.js 全局文件里，可以接收 object 对象参数，同时还可以指定微信小程序的生命周期函数、全局函数和全局数据。具体代码如下所示。

```
App({
  onLaunch: function() {//生命周期函数
    //调用 API 从本地缓存中获取数据
    var logs = wx.getStorageSync('logs') || []
    logs.unshift(Date.now())
    wx.setStorageSync('logs', logs)
  },

  getUserInfo: function(cb) {//全局函数
    var that = this
    if (this.globalData.userInfo) {
      typeof cb == "function" && cb(this.globalData.userInfo)
    } else {
      //调用登录接口
      wx.getUserInfo({
        withCredentials: false,
        success: function(res) {
          that.globalData.userInfo = res.userInfo
          typeof cb == "function" && cb(that.globalData.userInfo)
        }
      })
    }
  },

  globalData: {//全局数据
    userInfo: null
  }
})
```

微信小程序在 App()函数里可以使用下列生命周期函数。

（1）onLaunch：生命周期函数。监听小程序初始化，当小程序初始化完成时，会触发

onLaunch（全局只触发一次）。

（2）onShow：生命周期函数。监听小程序显示，当小程序启动，或从后台进入前台显示时，会触发 onShow。

（3）onHide：生命周期函数。监听小程序隐藏，当小程序从前台进入后台时，会触发 onHide。

（4）onError：错误监听函数。当小程序发生脚本错误，或者 API 调用失败时，会触发 onError 并带上错误信息。

（5）其他：开发者可以添加任意的函数或数据到 Object 参数中，用 this 可以访问。

微信小程序前后台定义：

当用户点击左上角关闭，或者按了设备 Home 键离开微信时，小程序并没有直接销毁，而是进入了后台；当再次进入微信或再次打开小程序时，又会从后台进入前台。需要注意的是：只有当小程序进入后台一定时间，或者系统资源占用过高时，才会被真正销毁。

微信小程序在 App()函数里定义的是全局函数和全局数据，因此在页面文件夹的 js 文件里可以直接使用。比如存在“pages/index/index”这样页面路径，在 index 文件夹下面就会有 index.js 文件，在 index.js 文件中就可以使用 App()函数里提供的全局函数或数据，在使用之前，需要使用 getApp()函数来获取小程序实例，这样就可以使用 App()函数提供的全局函数或数据，具体代码如下所示。

```
var app = getApp()
Page({
  onLoad: function () {
    console.log('onLoad')
    var that = this
    //调用应用实例的方法获取全局数据
    app.getUserInfo(function(userInfo){
      //更新数据
      that.setData({
        userInfo:userInfo
      })
    })
  }
})
```

注意：App()函数必须在 app.js 中注册，且不能注册多个。不要在定义 App()内的函数中调用 getApp()，使用 this 就可以拿到 App 实例。不要在 onLaunch 的时候调用 getCurrentPages()，此时 page 还没有生成。通过 getApp()获取实例之后，不要私自调用生命周期函数。

1.3 Page()函数使用

App()函数可用来注册一个微信小程序，同时提供生命周期函数、全局函数和全局数据，而 Page 函数则用来注册一个具体的页面，它在页面文件夹的 js 文件里，可以接受一个 object 参数，可以指定页面的初始数据、生命周期函数、事件处理函数等。

示例代码如下所示。

```
//index.js
Page({
  data: {
    text: "This is page data."
  },
  onLoad: function(options) {
    // Do some initialize when page load.
  },
  onReady: function() {
    // Do something when page ready.
  },
  onShow: function() {
    // Do something when page show.
  },
  onHide: function() {
    // Do something when page hide.
  },
  onUnload: function() {
    // Do something when page close.
  },
  onPullDownRefresh: function() {
    // Do something when pull down.
  },
  onReachBottom: function() {
    // Do something when page reach bottom.
  },
  onShareAppMessage: function () {
   // return custom share data when user share.
  },
  // Event handler.
  viewTap: function() {
    this.setData({
```

```
            text: 'Set some data for updating view.'
        })
    },
    customData: {
        hi: 'MINA'
    }
})
```

1.3.1　初始化数据

在 Page()函数的 data 对象里可以初始化页面要用到的函数，data 会以 JSON 的形式由逻辑层传至渲染层，所以其数据必须是可以转成 JSON 的格式：字符串、数字、布尔值、对象或数组。

假如界面需要姓名和性别的值，这时就可以通过 data 来传递数据，具体代码如下所示。

```
Page({
    data: {
        name: 'kevin',
        sex: '男'
    }
})
```

在页面里就可以把姓名和性别的值通过数据绑定的方式取出来，具体代码如下所示。

```
<view>{{name}}</view>
<view>{{sex}}</view>
```

1.3.2　生命周期函数

在 Page()函数里也提供了一些生命周期函数，如下所示。

（1）onLoad：生命周期函数。监听页面加载，一个页面只会调用一次。

（2）onReady：生命周期函数。监听页面初次渲染完成，一个页面只会调用一次，代表页面已经准备妥当，可以和视图层进行交互。

（3）onShow：生命周期函数。监听页面显示，每次打开页面都会调用一次。

（4）onHide：生命周期函数。监听页面隐藏，当 navigateTo 或底部 tab 切换时调用。

（5）onUnload：生命周期函数。监听页面卸载，当 redirectTo 或 navigateBack 时调用。

1.3.3 相关事件函数

在 Page()函数里也提供了一些常用的函数，比如下拉刷新、页面转发事件函数。

（1）onPullDownRefresh：下拉刷新。

监听用户下拉刷新事件，需要在 config 的 window 选项中开启 enablePullDownRefresh，当处理完数据刷新后，wx.stopPullDownRefresh 可以停止当前页面的下拉刷新。

（2）onShareAppMessage：用户转发。

只有定义了此事件的处理函数，右上角菜单才会显示“转发”按钮。用户点击转发按钮的时候会调用，此事件需要 return 一个 Object，用于自定义转发内容。

1.4 数据绑定

WXML 页面里的动态数据都是来自 js 文件 Page 的 data，数据绑定就是通过双大括号（{{}}）将变量包起来，在 WXML 页面里将数据值显示出来。

示例代码如下所示。

```
index.wxml
<view> {{ name }} </view>
```

```
index.js
Page({
  data: {
    name: 'kevin'
  }
})
```

（1）组件属性绑定是将 data 里的数据绑定到微信小程序的组件上，示例代码如下所示。

```
<view id="item-{{id}}"> </view>
```

```
Page({
  data: {
    id: 1
  }
})
```

（2）控制属性绑定用来进行 if 语句条件判断，如果满足条件，则执行，否则不执行，

示例代码如下所示。

```
<view wx:if="{{condition}}"> </view>
```

```
Page({
  data: {
    condition: true
  }
})
```

（3）关键字绑定常用于组件的一些关键字，像复选框组件一样，checked 关键字如果等于 true，则代表选中复选框，false 则代表不选中复选框，示例代码如下所示。

```
<checkbox checked="{{false}}"> </checkbox>
```

不要直接写 checked="false"，其计算结果是一个字符串，转成 boolean 类型后代表真值。

1.5　wx:if 条件判断

微信小程序的条件判断分为单个组件的条件判断和多个组件的条件判断。

1．wx:if 判断单个组件

在微信小程序里，可以将 wx:if="{{condition}}"应用到某个组件上，用{condition}}" 来判断是否需要渲染该代码块，示例代码如下所示。

```
<view wx:if="{{condition > 521}}"> True </view>
```

使用 `wx:elif` 和 `wx:else` 来添加一个 else 块。

```
<view wx:if="{{len > 50}}"> 1 </view>
<view wx:elif="{{len > 20}}"> 2 </view>
<view wx:else> 3 </view>
```

2．block wx:if 判断多个组件

因为 wx:if 是一个控制属性，因而它需要应用到具体的一个组件上。如果想一次性判断多个组件标签，那么这时可以使用一个 <block/> 标签将多个组件包装起来，并在上面使用 wx:if 控制属性，示例代码如下所示。

```
<block wx:if="{{true}}">
  <view> 姓名 </view>
  <view>性别 </view>
</block>
```

<block/> 并不是一个组件，它仅仅是一个包装元素，不会在页面中做任何渲染，只接受控制属性。

1.6 wx:for 列表渲染

列表渲染的使用和条件判断使用类似，也是分为应用到单个组件还是多个组件上。

1．wx:for 列表渲染单个组件

在组件上使用 wx:for 控制属性绑定一个数组，即使用数组中的各项数据重复渲染该组件。数组的当前项的下标变量名默认为 index，数组当前项的变量名默认为 item，示例代码如下所示。

```
<view wx:for="{{ students }}">
  {{index}}: {{item. name }}
</view>
```

```
Page({
  data: {
    students: [{
      name: 'kevin',
    }, {
      name: 'tom'
    }]
  }
})
```

使用 wx:for-item 可以指定数组当前元素的变量名，使用 wx:for-index 可以指定数组当前下标的变量名，示例代码如下所示。

```
<view wx:for="{{ students }}" wx:for-index="idx" wx:for-item="itemName">
  {{idx}}: {{itemName. name }}
</view>
```

2．block wx:for 列表渲染多个组件

wx:for 通常应用在某一个组件上，但是如果想渲染一个包含多节点的结构块，这时 wx:for 就需要应用在<block/>标签上，示例代码如下所示。

```
<block wx:for="{{[100, 200, 300]}}">
  <view> {{index}}: </view>
  <view> {{item}} </view>
</block>
```

1.7　定义和使用模板

微信小程序为了解决页面里重复代码的问题，提供了模板机制，把一些可以复用的代码放置在模板里进行定义，定义后就可以在界面里进行使用。WXML 提供模板（template），可以在模板中定义代码片段，然后在不同的地方调用。

1．定义模板

建立一个存放模板的页面文件，比如将文件命名为 template.wxml。template 模板是通过 name 属性来标识的，示例代码如下所示。

```
<!-- template.wxml -->
<template name="head">
  <text>{{userName}}</text>
</template>
```

这样就定义好了以 head 为名称的模板，然后就可以在其他页面进行使用了。

2．使用模板

定义了 head 为名称的模板，通过 import 引入到要使用模板的页面，在 WXML 页面文件里，使用 is 属性来查找 head 这个模板并进行使用，然后将模板所需要的 data 传入它需要传入 userName 的值，然后就可以在页面模板里渲染出来，示例代码如下所示。

```
<!--index.wxml→
<import src="template.wxml"/>
<template is="head" data="{{item}}"/>
```

```
//index.js
Page({
  data: {
    item: {
      userName: kevin
    }
  }
})
```

模板的 is 属性还支持三元运算，示例代码如下所示。

```
<template name="red">
  <view>我是红色 </view>
</template>
<template name="blue">
  <view>我是蓝色</view>
```

```
</template>

<block wx:for="{{[1, 2, 3, 4]}}">
      <template is="{{item % 2 == 0 ? red: blue }}"/>
</block>
```

1.8　import 引用模板

import 引用可以将有模板功能的页面文件引入到要使用模板的页面里，也就是说，要使用模板功能，就必须先通过 import 引入模板文件，然后才能找到模板进行使用。

1．import 引用模板

在 template.wxml 中定义了一个叫 person 的 template，示例代码如下所示。

```
<!-- template.wxml -->
<template name="person">
   <text>姓名：{{name}}</text>
   <text>年龄：{{age}}</text>
   <text>性别：{{sex}}</text>
</template>
```

在 index.wxml 中引用了 template.wxml，就可以使用 person 模板，示例代码如下所示。

```
<import src="item.wxml"/>
<template is=" person " data="{{ name: 'kevin',age:'12',sex:'男'}}"/>
```

只有通过 import 引入模板文件，才能使用模板功能。

2．import 引用作用域

import 有作用域的概念，即只会 import 目标文件中定义的 template，而不会 import 目标文件 import 的 template。

比如在 second.wxml 中可以使用 first.wxml 里的模板，因为 second.wxml 文件通过 import 引入 first.wxml 文件模板；在 third 文件中可以使用 second 里的模板，因为 third.wxml 文件通过 import 引入 second.wxml 文件模板，但是不可以使用 first.wxml 文件模板，因为没有引入 first.wxml 文件。

```
<!-- first.wxml -->
<template name="first">
   <text>第一个模板</text>
</template>
```

```
<!-- second.wxml -->
<import src="first.wxml"/>
<template name="second">
  <text>第二个模板</text>
</template>

<!-- third.wxml -->
<import src="second.wxml"/>
<template is="first"/>   <!-- Error! Can not use tempalte when not import first. -->
<template is="second"/>
```

1.9　include 引用文件

include 的引用功能是引用除 template 模板代码之外的全部代码，相当于把引用文件的代码复制过来。

比如每个页面都有一样的导航菜单和版权信息，这时就可以把导航菜单代码和版权信息代码都放置在一个单独的文件中，然后就可以在页面中直接使用了，并不需要重复地编写代码，同时能保持代码、页面展现的一致性。

```
<!-- navigator.wxml -->
<view>推荐</view>
<view>科技</view>
<view>军事</view>
<view>娱乐</view>
<view>美食</view>
<view>游戏</view>

<!-- footer.wxml -->
<view> 版权：归小刚所有 </view>


<!-- index.wxml -->
<include src=" navigator.wxml"/>
<view>我是正文内容 </view>
<include src="footer.wxml"/>
```

1.10　Console 面板

Console 面板可用来在开发过程中输出调试信息的，如果代码编写有误，则会输出相应的错误，如图 1.8 所示。

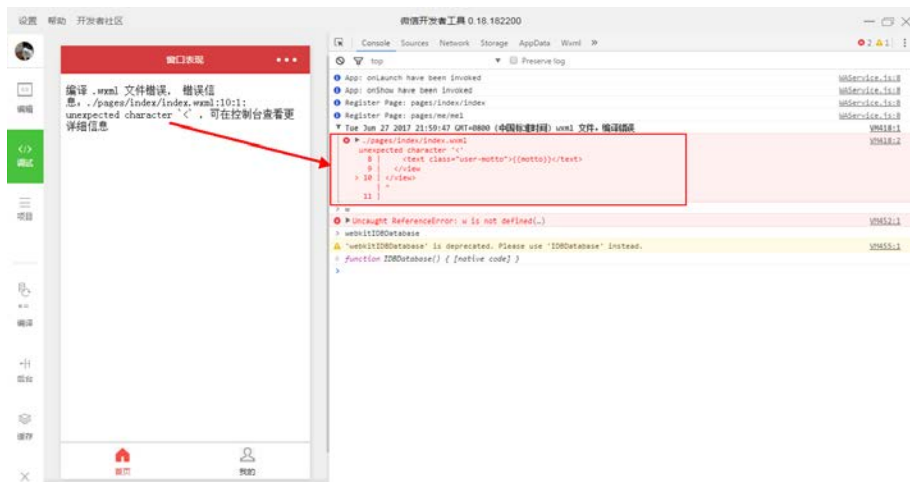


图 1.8　输出错误信息

除了可以输出错误信息，同时也允许输入代码，如图 1.9 所示。

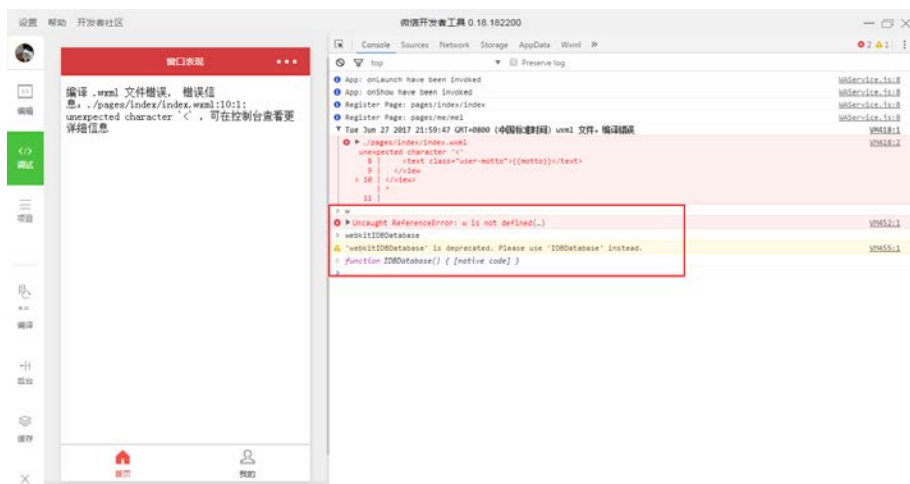


图 1.9　输入代码

1.11　Sources 面板

Sources 面板可用来显示当前项目的脚本文件，它包括两部分脚本文件：一部分是开发者自己编写的脚本文件，另一部分是工具根据开发者的脚本进行包装的脚本文件，开发者的代码都会被包裹在 define 函数中，并且对于 Page 代码，在尾部会有 require 的主动调用，如图 1.10 和图 1.11 所示。

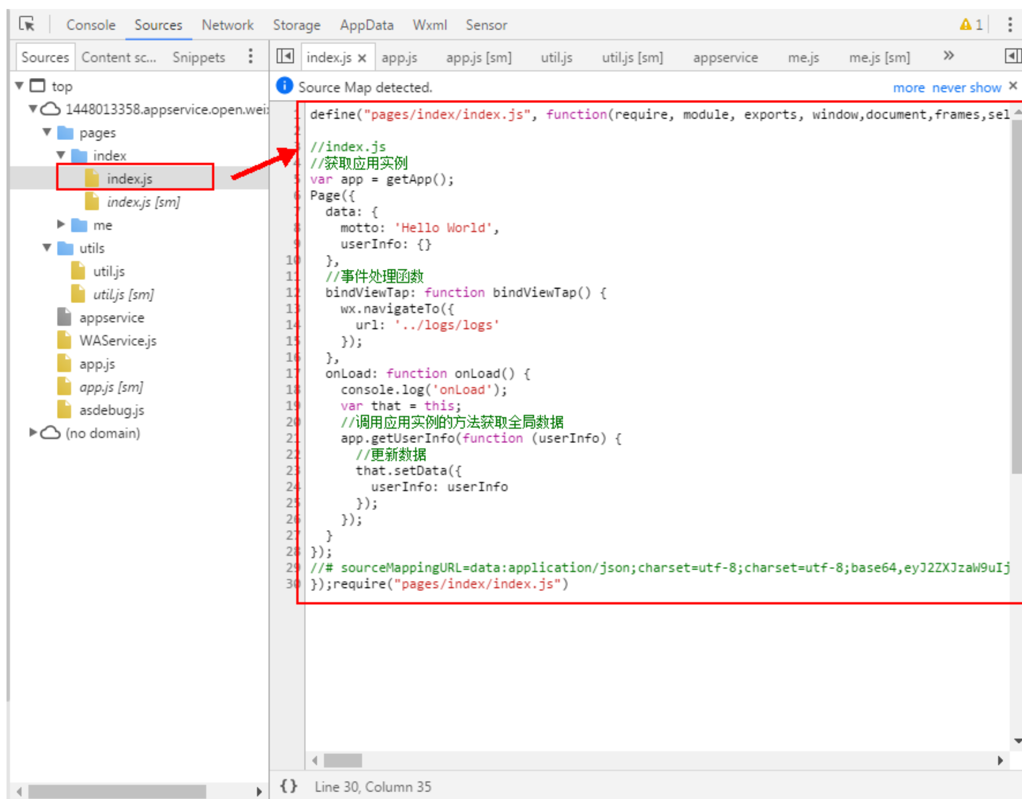


图 1.10　包装的脚本文件

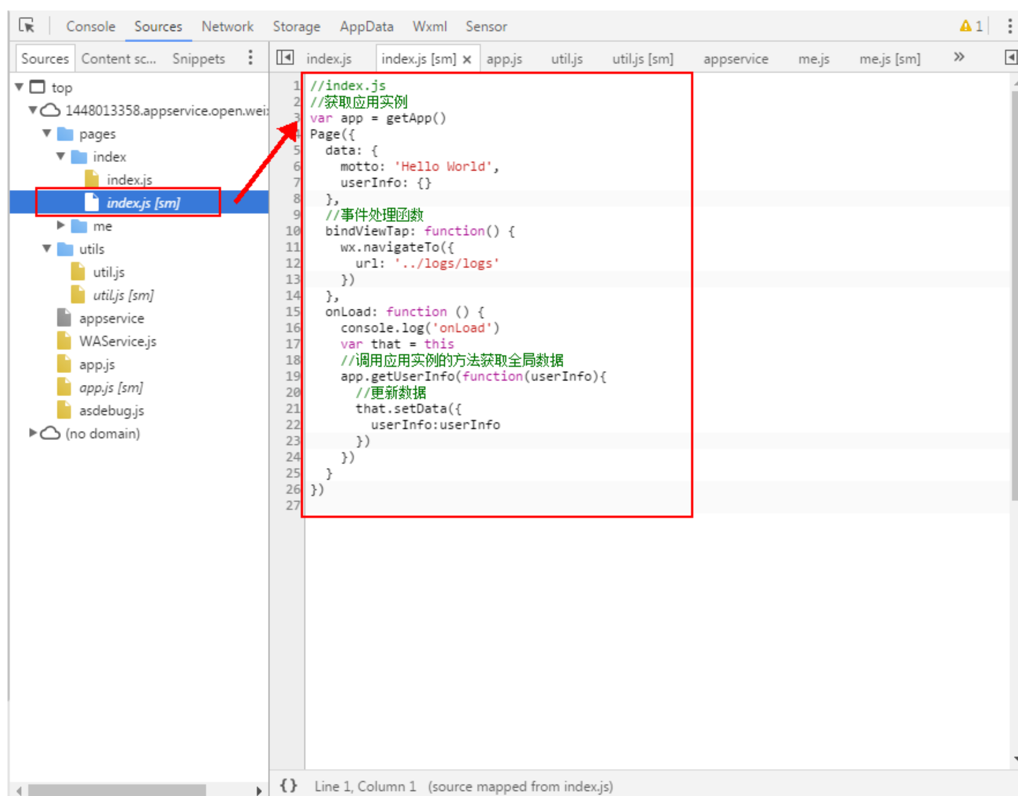


图 1.11　原生的脚本文件

1.12　Network 面板

Network 面板可用来显示调用哪些文件以及这些文件调用的状态，可以看到所有调用的文件信息，包括文件名称、路径、大小、调用的状态、时间等信息，如图 1.12 所示。

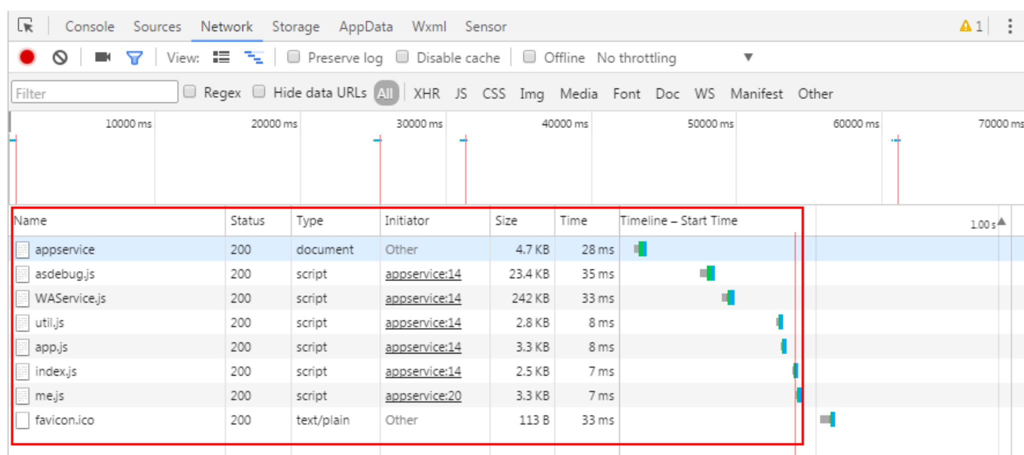


图 1.12　调用文件

单击某一个调用的文件，会查看到 HTTP 的请求，包括请求的 URL、请求的方式（GET/POST）、请求的状态（200 代表成功）、IP 地址等，如图 1.13 所示。

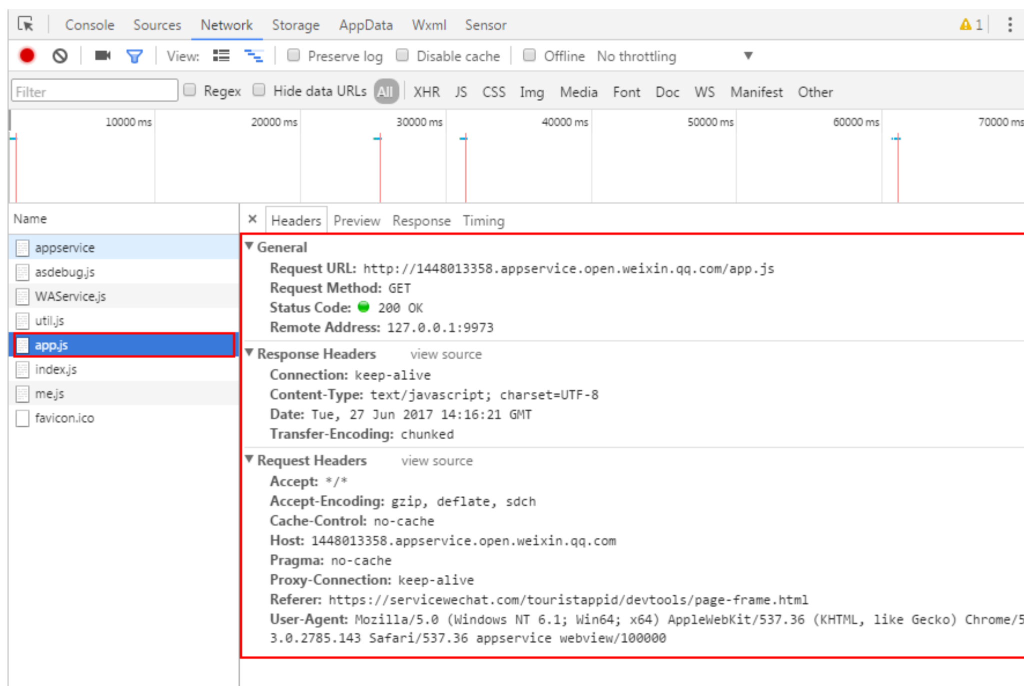


图 1.13　HTTP 请求

1.13　Storage 面板

Storage 面板可用来管理本地缓存数据的地方，可以使用 wx.setStorage 或者 wx.setStorageSync 这两个 API 将数据缓存起来，在 Storage 面板中就可以查看，如图 1.14 所示。

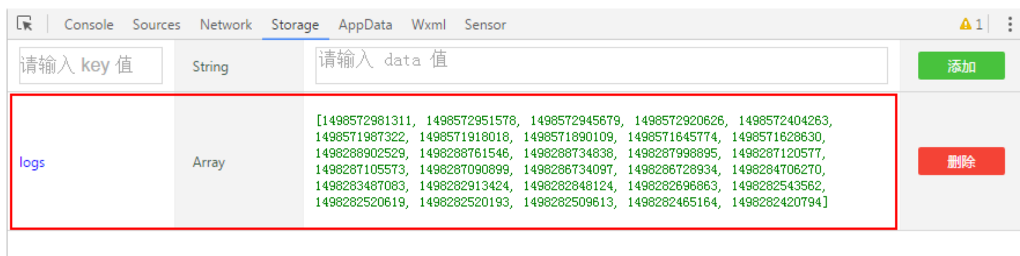


图 1.14　logs 本地缓存数据

在 Storage 面板里可以对缓存数据进行管理，既可以删除缓存的本地数据，也可以在 Storage 面板里通过 key-value 的形式添加缓存数据，如图 1.15 所示。

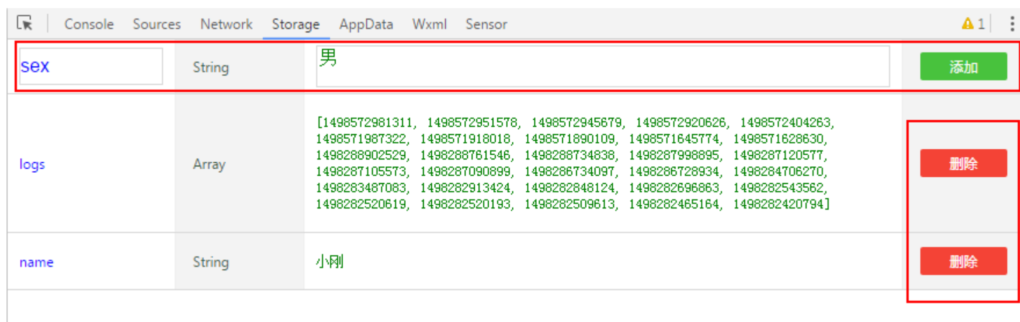


图 1.15　添加/删除本地缓存数据

1.14　Appdata 面板

Appdata 面板可用来显示当前页面当前时刻的数据，并且可以查看本页面用到了哪些数据，如图 1.16 所示。

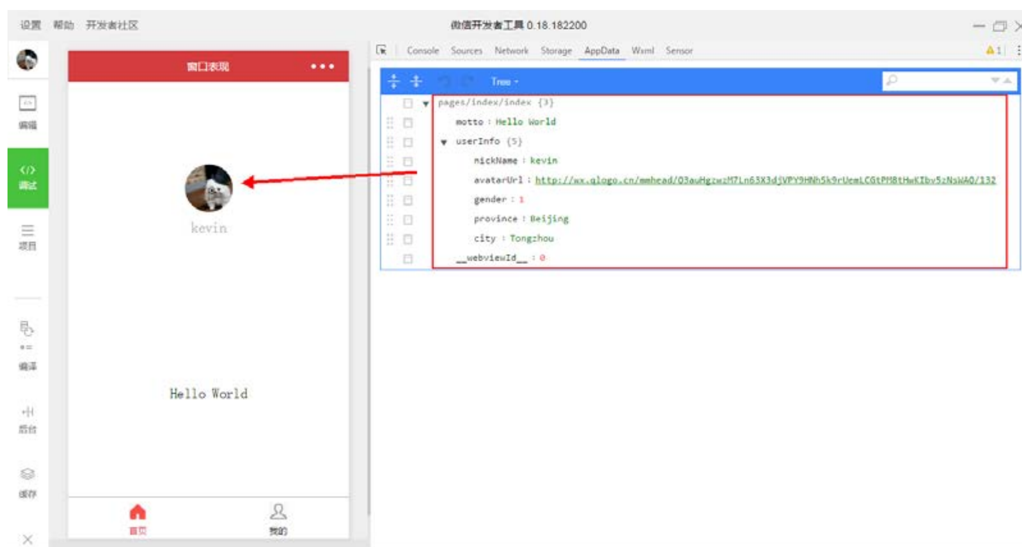


图 1.16　当前页面数据

除了显示当前页面的数据，还可以对当前页面数据进行编辑，修改当前页面的数据，如图 1.17 所示。

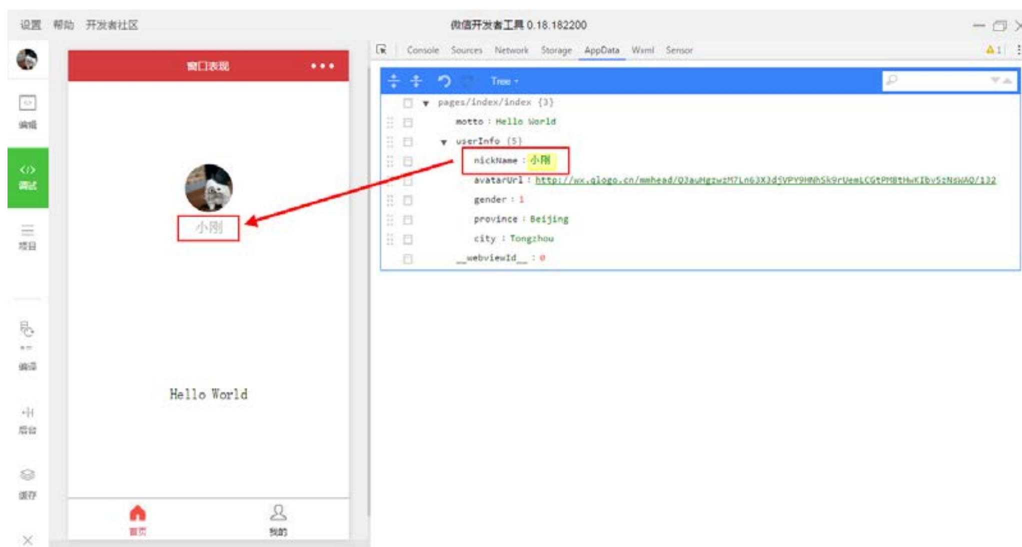


图 1.17　修改页面数据

1.15　Wxml 面板

Wxml 面板可用来显示当前页面的代码以及展示的数据，如图 1.18 所示。

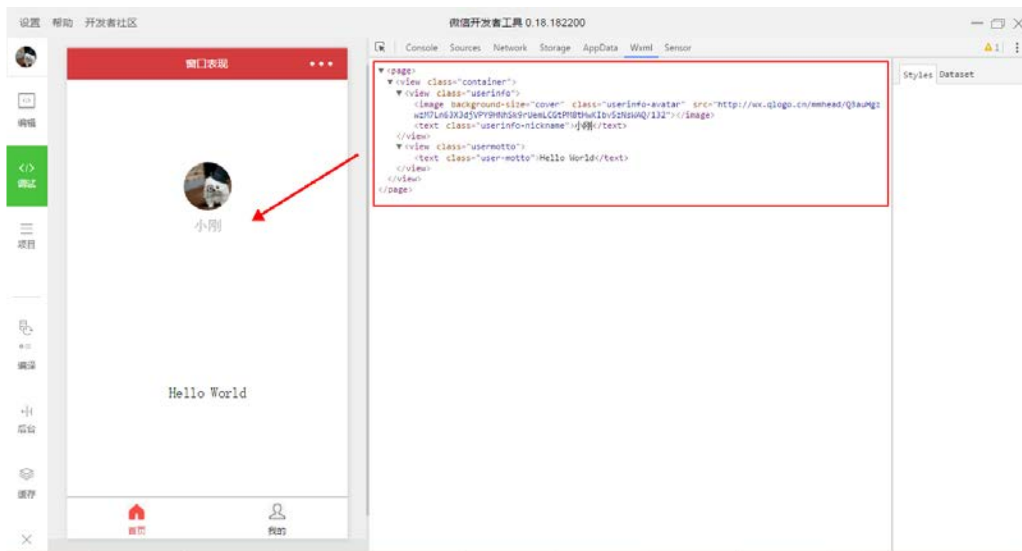


图 1.18　当前页面代码和数据

同时还可以编辑代码的样式，方便进行界面的布局和样式的编写，如图 1.19 所示。

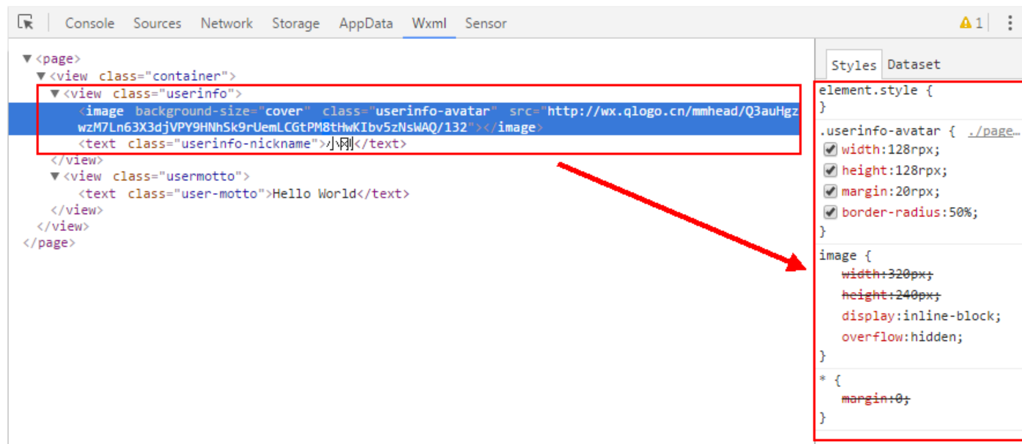


图 1.19　编写样式

1.16　Sensor 面板

Sensor 面板可用来选择模拟地理位置，如图 1.20 所示。

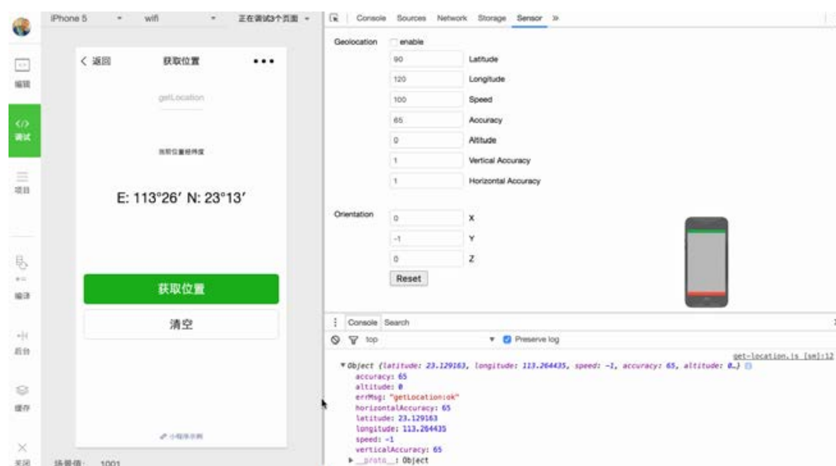


图 1.20　模拟地理位置

除了可以模拟地理位置，还可以模拟移动设备表现，用于调试重力感应 API，如图 1.21 所示。

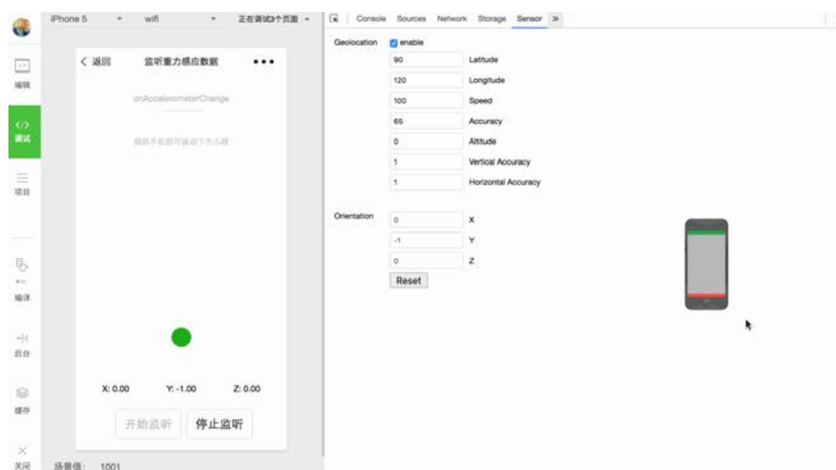


图 1.21　模拟重力感应

第 2 章　微信小程序丰富的组件

微信小程序提供了丰富的组件，可以分为八类：视图容器类组件、基础内容类组件、表单类组件、导航类组件、媒体类组件、地图类组件、画布类组件和客服类组件。

（1）视图类组件：view 视图容器组件、scroll-view 可滚动视图容器组件、swiper 滑块视图容器组件、movable-area 可移动的视图容器组件。

（2）基础内容类组件：icon 图标组件、text 文本组件、progress 进度条组件。

（3）表单类组件：button 按钮组件、checkbox 多项选择器组件、form 表单组件、input 输入框组件、label 改进表单组件、picker 选择器组件、picker-view 嵌入页面的滚动选择器、radio 单项选择器组件、slider 滑动选择器组件、switch 开关选择器组件、textarea 多行输入框组件。

（4）导航类组件：navigator 页面链接组件。

（5）媒体类组件：audio 音频组件、image 图片组件、video 视频组件。

（6）地图类组件：map 地图组件。

（7）画布类组件：canvas 画布组件。

（8）客服类组件：contact-button 客服会话按钮组件。

2.1　view 视图容器组件

view 视图容器组件是最基本的页面布局会用到的组件，它的使用类似网页开发的 div 组件，通过 view 视图容器组件的布局及样式的编写，就可以实现想要的页面效果。在进行微信小程序开发的时候，经常会设计列表式导航，这时就可以用 view 视图容器组件来完成列表式导航的布局设计，如图 2.1 所示。



图 2.1　列表式导航

（1）在 view.wxml 页面文件里，使用 view 视图容器组件进行页面布局，具体代码如下所示。

```
<view class="content">
  <view class="hr"></view>
  <view class="item">
    <view class="order">消息通知</view>
    <view class="detail">
      <text>></text>
    </view>
  </view>
  <view class="hr"></view>
  <view class="item">
    <view class="order">头条商城</view>
    <view class="detail">
      <text>点击速领 200 元新年红包  ></text>
    </view>
  </view>
  <view class="xian"></view>
  <view class="item">
    <view class="order">京东特供</view>
    <view class="detail">
      <text>></text>
    </view>
  </view>
  <view class="hr"></view>
  <view class="item">
```

```
    <view class="order">我要爆料</view>
    <view class="detail">
      <text>></text>
    </view>
  </view>
  <view class="xian"></view>
  <view class="item">
    <view class="order">用户反馈</view>
    <view class="detail">
      <text>></text>
    </view>
  </view>
  <view class="xian"></view>
  <view class="item" bindtap="setting">
    <view class="order">系统设置</view>
    <view class="detail">
      <text>></text>
    </view>
  </view>
  <view class="xian"></view>
</view>
```

（2）在 view.wxss 样式文件里，给 view.wxml 页面文件添加样式，具体代码如下所示。

```
.bg{
  width:100%;
  height: 150px;
  background-color: #D53E37;
}
.hr{
    width: 100%;
    height: 15px;
    background-color: #F4F5F6;
}
.item{
    display:flex;
    flex-direction:row;
}
.order{
    padding-top:15px;
    padding-left: 15px;
    padding-bottom:15px;
```

```
    font-size:15px;
}
.detail{
    font-size: 15px;
    position: absolute;
    right: 10px;
    height: 50px;
    line-height: 50px;
    color: #888888;
}
.xian{
    border: 1px solid #cccccc;
    opacity: 0.2;
}
```

这样就可以完成列表式导航设计，通过 view 视图容器组件的使用以及添加样式，就可以实现想要的效果。

2.2　scroll-view 可滚动视图容器组件

view 视图容器组件是不可以滚动的，如果想实现滚动的效果，就需要使用 scroll-view 可滚动视图容器组件。scroll-view 可滚动视图容器组件的属性如表 2.1 所示。

表 2.1　scroll-view可滚动视图容器组件的属性

属 性 名	类　　型	默 认 值	说　　明
scroll-x	Boolean	false	允许横向滚动
scroll-y	Boolean	false	允许纵向滚动
upper-threshold	Number	50	距顶部/左边多远时（单位px），触发scrolltoupper事件
lower-threshold	Number	50	距底部/右边多远时（单位px），触发scrolltolower事件
scroll-top	Number		设置竖向滚动条位置
scroll-left	Number		设置横向滚动条位置
scroll-into-view	String	String	值应为某子元素id（id不能以数字开头）。设置哪个方向可滚动，则在哪个方向滚动到该元素
scroll-with-animation	Boolean	false	在设置滚动条位置时使用动画过渡
enable-back-to-top	Boolean	false	iOS点击顶部状态栏、安卓双击标题栏时，滚动条返回顶部，只支持竖向

续表

属性名	类　型	默认值	说　明
bindscrolltoupper	EventHandle		滚动到顶部/左边，会触发scrolltoupper事件
bindscrolltolower	EventHandle		滚动到底部/右边，会触发scrolltolower事件
bindscroll	EventHandle		滚动时触发，event.detail = {scrollLeft, scrollTop, scrollHeight, scrollWidth, deltaX, deltaY}

经常会用到 scroll-x 设置允许横向滚动，scroll-y 允许纵向滚动，使用竖向滚动时，需要给<scroll-view/>一个固定高度，通过 WXSS 设置 height。

使用过腾讯新闻 App 和今日头条 App 来阅读新闻的用户，都会发现这样一个功能，它的新闻频道可以左右滑动，然后单击自己喜欢的新闻频道就可以阅读相关的新闻，如图 2.2 所示。



图 2.2　新闻频道

（1）在 scroll-view.wxml 页面文件里，可以使用 view、scroll-view 进行新闻频道的布局，具体代码如下所示。

```
<view class="content">
<view class="navbg">
    <view class="nav">
        <scroll-view class="scroll-view_H" scroll-x="true" >
        <view class="scroll-view_H">
            <view><view class="{{flag==0?'select':'normal'}}" id="0" bindtap="switchNav">推荐</view></view>
            <view><view class="{{flag==1?'select':'normal'}}" id="1" bindtap="switchNav">热点</view></view>
            <view><view class="{{flag==2?'select':'normal'}}" id="2" bindtap="switchNav">北京</view></view>
            <view><view class="{{flag==3?'select':'normal'}}" id="3" bindtap="switchNav">社会</view></view>
            <view><view class="{{flag==4?'select':'normal'}}" id="4" bindtap="switchNav">娱乐</view></view>
            <view><view class="{{flag==5?'select':'normal'}}" id="5" bindtap="switchNav">问答</view></view>
            <view><view class="{{flag==6?'select':'normal'}}" id="6" bindtap="switchNav">图片</view></view>
```

```
            <view><view  class="{{flag==6?'select':'normal'}}"  id="7"  bindtap="switchNav"> 科 技
</view></view>
            <view><view  class="{{flag==6?'select':'normal'}}"  id="8"  bindtap="switchNav"> 汽 车
</view></view>
            <view><view  class="{{flag==6?'select':'normal'}}"  id="9"  bindtap="switchNav"> 体 育
</view></view>
        </view>
  </scroll-view>
   </view>
   <view class="add">+</view>
</view>
</view>
```

（2）在 scroll-view.wxss 样式文件里，给新闻频道添加样式，具体代码如下所示。

```
.bg{
    background-color: #D53C3E;
    height: 40px;
    display: flex;
    flex-direction:row;
}
.navbg{
  background-color: #F6F5F3;
  height: 36px;
  color: #000000;
  display: flex;
  flex-direction: row;
  align-items: center;
}
.nav{
  width: 85%;
  height: 36px;
}
.add{
  width: 15%;
  height: 50px;
  line-height: 50px;
  text-align: right;
  margin-right:10px;
  font-size: 50px;
}
.scroll-view_H{
```

```
    height: 40px;
    display: flex;
    flex-direction: row;
    margin-left: 5px;
}
.normal{
    width: 40px;
    height: 40px;
    line-height: 40px;
    padding-left:5px;
    padding-right: 5px;
    font-size: 14px;
}
.select{
      width: 40px;
    height: 40px;
    line-height: 40px;
    padding-left:5px;
    padding-right: 5px;
    font-size: 14px;
    font-weight: bold;
}
```

（3）在 scroll-view.js 文件里，定义 flag 变量，默认值为 0，代表当前选中的是第一个新闻频道，具体代码如下所示。

```
Page({
    data: {
        flag: 0
    }
})
```

这样就可以实现新闻频道的左右滑动效果，通过使用 scroll-view 可滚动视图容器组件的 scroll-x="true"来设置允许横向滑动效果。

2.3 swiper 滑块视图容器组件

swiper 滑块视图容器组件使用得非常频繁，这个组件有两种常用的效果：一种是海报轮播效果，另一种是页签切换效果，它的属性如表 2.2 所示。

表 2.2　swiper滑块视图容器组件属性

属 性 名	类　　型	默 认 值	说　　明
indicator-dots	Boolean	false	是否显示面板指示点
indicator-color	Color	rgba(0, 0, 0, .3)	指示点颜色
indicator-active-color	Color	#000000	当前选中的指示点颜色
autoplay	Boolean	false	是否自动切换
current	Number	0	当前所在页面的index
interval	Number	5000	自动切换时间间隔
duration	Number	500	滑动动画时长
circular	Boolean	false	是否采用衔接滑动
bindchange	EventHandle		current 改变时会触发 change 事件，event.detail = {current: current}

swiper 滑块视图容器组件内只能放置<swiper-item>组件，否则会导致未定义的行为。

2.3.1　海报轮播效果

海报轮播效果是很多 App 软件或者网站会采用的一种布局方式，它可以在有限区域，通过海报轮播的方式，动态地显示不同广告信息或者推广信息，如图 2.3 所示。

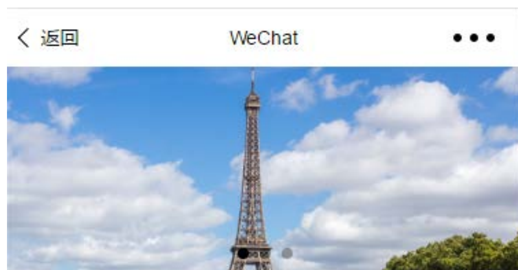


图 2.3　海报轮播效果

（1）在 swiper.wxml 页面文件里进行海报轮播效果的布局，具体代码如下所示。

```
<view  class="haibao">
    <swiper  indicator-dots="{{indicatorDots}}"  autoplay="{{autoplay}}"  interval="{{interval}}"  duration="{{duration}}">
        <block  wx:for="{{imgUrls}}">
            <swiper-item>
```

```
            <image src="{{item}}" class="silde-image"></image>
          </swiper-item>
       </block>
    </swiper>
 </view>
```

（2）在 swiper.wxss 样式文件里，给海报轮播区域添加样式，具体代码如下所示。

```
.haibao{
    text-align: center;
    width: 100%;
}
.silde-image{
    width: 100%;
}
```

（3）在 swiper.js 样式文件里，海报轮播区域需要的数据必须在 data 里进行初始化，具体代码如下所示。

```
Page({
  data: {
    indicatorDots: true,
    autoplay: true,
    interval: 5000,
    duration: 1000,
    imgUrls: [
      "http://img02.tooopen.com/images/20150928/tooopen_sy_143912755726.jpg",
      "http://img06.tooopen.com/images/20160818/tooopen_sy_175866434296.jpg",
      "http://img06.tooopen.com/images/20160818/tooopen_sy_175833047715.jpg"
    ]
  }
})
```

这样就可以使用 swiper 滑块视图容器组件来完成海报轮播效果了，同时还可以设置是否显示面板指示点、指示点颜色、当前选中的指示点颜色、是否自动切换、自动切换时间间隔、滑动动画时长等属性。

2.3.2 页签切换效果

使用 swiper 滑块视图容器组件也可以完成页签切换效果，页签切换效果也是很多 App 软件和微信小程序经常会用到的一种操作效果。现在很多网站登录都有多种方式，比如账号密码登录和手机快捷登录，这时就可以使用 swiper 滑块视图容器组件在这两种方式间进

行切换，如图 2.4 和图 2.5 所示。



图 2.4　账号密码登录



图 2.5　手机快捷登录

（1）在 swiper.wxml 文件里，使用 view、swiper 组件实现两种登录方式的布局设计，具体代码如下所示。

```
<view class="content">
   <view class="loginTitle">
     <view class="{{currentTab==0?'select':'default'}}" data-current="0" bindtap="switchNav">账号密码登录</view>
     <view class="{{currentTab==1?'select':'default'}}" data-current="1" bindtap="switchNav">手机快捷登录</view>
   </view>
   <view class="hr"></view>
   <swiper current="{{currentTab}}"style="height:{{winHeight}}px">
      <swiper-item>
         <view class="accountType">
            <view class="account">
               <view class="ac">账号</view>
               <view class="ipt"><input name="account" bindblur="accountBlur" type="text" placeholder="手机/邮箱/用户名" class="placeholder-style"/></view>
            </view>
            <view class="hr"></view>
            <view class="account">
               <view class="ac">密码</view>
               <view class="ipt"><input name="password" bindblur="passwordBlur" type="te
```

```
xt" password placeholder="请输入密码" class="placeholder-style"/></view>
                </view>
                <view class="hr"></view>

            </view>
        </swiper-item>
        <swiper-item>
            <view class="mobileType">
                <view class="account">
                     <view class="ac">手机号</view>
                     <view class="ipt"><input name="mobile" bindblur="accountBlur" type="text" p
laceholder="仅支持中国大陆手机号" class="placeholder-style"/>
                     <button class="btn">获取验证码</button>
                     </view>
                </view>
                 <view class="hr"></view>
                 <view class="account">
                     <view class="ac">验证码</view>
                     <view class="ipt"><input name="code" bindblur="passwordBlur" type="text" p
assword placeholder="请输入验证码" class="placeholder-style"/></view>
                </view>
                <view class="hr"></view>
            </view>
        </swiper-item>
    </swiper>
</view>
```

（2）在 swiper.wxss 样式文件里，给这两种登录方式添加样式，具体代码如下所示。

```
.loginTitle{
     display: flex;
     flex-direction: row;
     width: 100%;
}
.select{
     font-size:12px;
     color: red;
     width: 50%;
     text-align: center;
     height: 45px;
     line-height: 45px;
     border-bottom:5rpx solid red;
```

```
}
.default{
      font-size:12px;
      margin: 0 auto;
      padding: 15px;
}
.hr{
      border: 1px solid #cccccc;
      opacity: 0.2;
}
.account{
      display: flex;
      flex-direction: row;
}
.ac{
      padding:15px;
      font-size:14px;
      font-weight: bold;
      color: #666666;
}
.ipt{
      padding-top:10px;
}
.ipt input{
      text-align: left;
      width: 200px;
}
.placeholder-style{
      font-size: 14px;
      color: #cccccc;
}
.btn{
      position: absolute;
      right: 10px;
      top: 10px;
      width: 90px;
      font-size: 12px;
      color: #666666;
      background-color: #f2f2f2;
}
```

（3）在 swiper.js 业务逻辑处理文件里，初始化相应的数据并添加页签切换的函数，具体代码如下所示。

```
Page({
  data: {
    currentTab: 0,
    winWidth: 0,
    winHeight: 0
  },
  onLoad: function (options) {
    var page = this;
    wx.getSystemInfo({
      success: function (res) {
        console.log(res);
        page.setData({ winWidth: res.windowWidth });
        page.setData({ winHeight: res.windowHeight });
      }
    })
  },
  switchNav: function (e) {
    var page = this;
    if (this.data.currentTab == e.target.dataset.current) {
      return false;
    } else {
      page.setData({ currentTab: e.target.dataset.current });
    }
  }
})
```

这样就可以实现两种登录方式的切换了，在实际项目中经常会遇到这种页签切换的需求，都可以采用同样的方式来实现。

2.4 movable-view 可移动的视图容器组件

movable-view 可移动的视图容器组件是用来实现组件移动效果的，在页面中可以通过拖曳的方式进行滑动，滑动时必须在 movable-area 区域内。必须给 movable-area 设置 width 和 height 属性，若不设置则默认为 10px。movable-view 可移动的视图容器组件的属性如表 2.3 所示。

表 2.3 movable-area可移动的视图容器组件的属性

属性名	类型	默认值	说明
direction	String	none	movable-view的移动方向，属性值有all、vertical、horizontal、none
inertia	Boolean	false	movable-view是否带有惯性
out-of-bounds	Boolean	false	超过可移动区域后，movable-view是否还可以移动
x	Number		定义X轴方向的偏移，如果x的值不在可移动范围内，则会自动移动到可移动范围；改变x的值会触发动画
y	Number		定义Y轴方向的偏移，如果y的值不在可移动范围内，则会自动移动到可移动范围；改变y的值会触发动画
damping	Number	20	阻尼系数，用于控制x或y改变时的动画和过界回弹的动画，值越大移动越快
friction	Number	2	摩擦系数，用于控制惯性滑动的动画，值越大摩擦力越大，滑动越快停止；必须大于0，否则会被设置成默认值

1．单个 movable-view 可移动的视图容器滑动

用 movable-area 设定一个宽度为 100%，高度为 200px 的一个可拖动灰色区域，在这个区域内放置一个大小为 50×50、红色的、可以拖动的 movable-view 可移动视图容器组件，将 movable-view 这个可移动的视图容器组件的 direction 设置为 all，表示可以在任意方向上进行拖动，如图 2.6 所示。

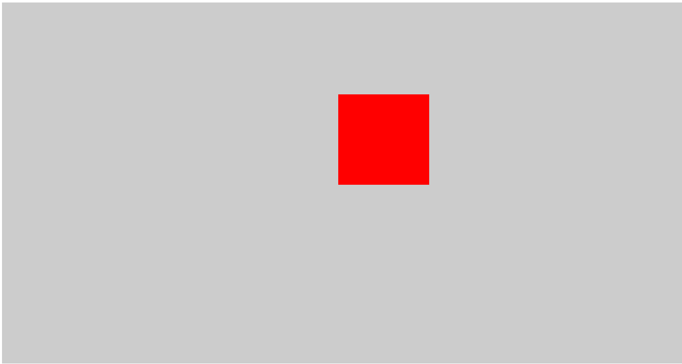


图 2.6 单个可移动视图容器组件

在 movable-view.wxml 里使用 movable-area 和 movable-view 来进行页面布局，具体代码如下所示。

```
<movable-area style="height: 200px;width: 100%;background:#cccccc;">
  <movable-view direction="all" style="height: 50px; width: 50px; background: red;">
```

```
  </movable-view>
</movable-area>
```

2．多个 movable-view 可移动的视图容器滑动

用 movable-area 设定一个宽度为 100%，高度为 200px 的可拖动灰色区域。在这个区域内放置一个大小为 50×50、红色的、可以拖动的 movable-view 可移动视图容器组件，将其 direction 设置为 all，表示可以在任意方向上进行拖动。再放置一个大小为 30×30、蓝色的、可以拖动的 movable-view 可移动视图容器组件，将其 direction 设置为 horizontal，表示在水平方向上可以拖动，如图 2.7 所示。

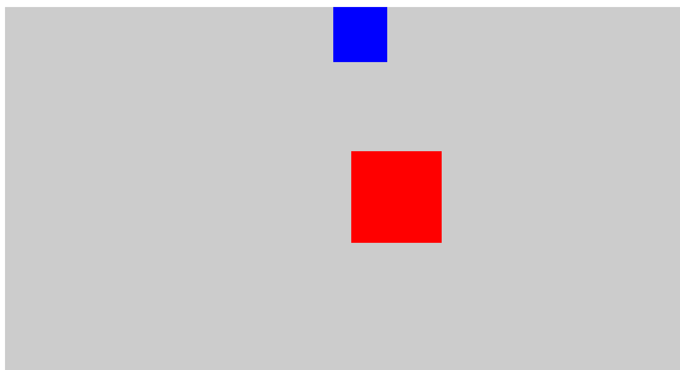


图 2.7　多个可移动视图容器组件

在 movable-view.wxml 里使用 movable-area 和 movable-view 来进行页面布局，具体代码如下所示。

```
<movable-area style="height: 200px;width: 100%;background: #cccccc;">
    <movable-view direction="all" style="height: 50px; width: 50px; background: red;">
    </movable-view>
    <movable-view direction="horizontal" style="height: 30px; width: 30px; background: blue;">
    </movable-view>
</movable-area>
```

2.5　icon 图标组件

微信小程序的 icon 图标组件可以用来设置图标类型、图标大小和图标颜色，icon 图标组件的属性如表 2.4 所示。

表 2.4　icon图标组件的属性

属 性 名	类　　型	默 认 值	说　　明
type	String		icon的类型，有效值：success、success_no_circle、info、warn、waiting、cancel、download、search、clear
size	Number	23	icon的大小，单位为px
color	Color		icon的颜色，同CSS的color

1．不同大小的图标组件

图标的类型和颜色一样，但是图标的大小不一样，如图 2.8 所示。



图 2.8　不同大小的图标组件

实现代码如下所示。

```
<view class="group">
  <block wx:for="{{[20, 30, 40, 50, 60, 70]}}">
    <icon type="success" size="{{item}}"/>
  </block>
</view>
```

2．不同类型的图标组件

图标的大小一样，但是图标的类型不一样，有 success、success_no_circle、info、warn、waiting、cancel、download、search 和 clear 这些类型，如图 2.9 所示。

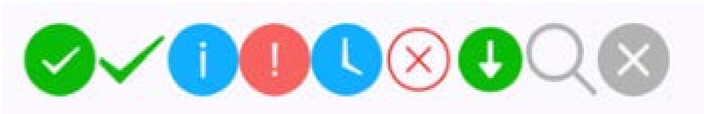


图 2.9　不同类型的图标组件

实现代码如下所示。

```
<view class="group">
  <block wx:for="{{['success', 'success_no_circle', 'info', 'warn', 'waiting', 'cancel', 'download',
'search', 'clear']}}">
    <icon type="{{item}}" size="40"/>
```

```
    </block>
</view>
```

3．不同颜色的图标组件

图标的类型和图标的大小一样，但是图标的颜色不一样，如图 2.10 所示。

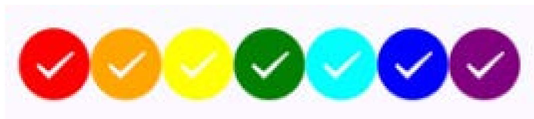


图 2.10　不同颜色的图标组件

实现代码如下所示。

```
<view class="group">
    <block wx:for="{{['red', 'orange', 'yellow', 'green', 'rgb(0,255,255)', 'blue', 'purple']}}">
        <icon type="success" size="40" color="{{item}}"/>
    </block>
</view>
```

2.6　text 文本组件

text 文本组件支持转义符“\”。<text/> 组件内只支持 <text/> 嵌套。它有 selectable 一个属性，代表文本是否可选。当 selectable 是可选的时候，除文本节点外，只其他节点都无法长按选中。

下面演示一下 text 文本组件支持转义符"\"，并且可以动态添加一行和删除一行，如图 2.11 所示。



图 2.11　text 文本组件应用

（1）在 text.wxml 页面文件里，使用 view、text、button 组件进行布局，具体代码如下所示。

```
<view class="btn-area">
  <view class="body-view">
    <text>{{text}}</text>
    <button bindtap="add">添加一行</button>
    <button bindtap="remove">删除一行</button>
  </view>
</view>
```

（2）在 text.js 业务逻辑处理文件里，添加页面初始化数据、新增行数据和移除行数据，具体代码如下所示。

```
var initData = '这是第一行\n 这是第二行'
var extraLine = [];
Page({
  data: {
    text: initData
  },
  add: function (e) {
    extraLine.push('新增一行')
    this.setData({
      text: initData + '\n' + extraLine.join('\n')
    })
  },
  remove: function (e) {
    if (extraLine.length > 0) {
      extraLine.pop()
      this.setData({
        text: initData + '\n' + extraLine.join('\n')
      })
    }
  }
})
```

2.7　progress 进度条组件

微信小程序的 progress 进度条组件也是经常会用到的一个组件，可以设置进度条的百分比、进度条的宽度和进度条的颜色，具体属性如表 2.5 所示。

表 2.5　progress进度条组件的属性

属 性 名	类　　型	默 认 值	说　　明
percent	Float		百分比0~100
show-info	Boolean	false	在进度条右侧显示百分比
stroke-width	Number	6	进度条线的宽度，单位为px
color	Color	#09BB07	进度条颜色（请使用 activeColor）
activeColor	Color		已选择的进度条颜色
backgroundColor	Color		未选择的进度条颜色
active	Boolean	false	进度条从左往右的动画

制作一个进度条，未选择颜色为黑色，已选择颜色为红色，进度条线的宽度为 30px，进度条有从左往右的动画效果，显示百分比为 60%，如图 2.12 所示。



图 2.12　进度条

实现代码如下所示。

```
<progress percent="60" stroke-width="30" show-info active  backgroundColor="black" activeC
olor="red"/>
```

2.8　button 按钮组件

微信小程序的 button 按钮组件是进行表单操作时经常会用到的一个组件，它可以用来进行表单的提交或者重置表单，同时 button 按钮组件通过不同的属性设置，可以拥有不同样式不同效果的 button 按钮，button 按钮组件的属性如表 2.6 所示。

表 2.6　button按钮组件的属性

属 性 名	类　　型	默 认 值	说　　明
size	String	default	按钮的大小
type	String	default	按钮的样式类型
plain	Boolean	false	按钮是否镂空，背景色透明
disabled	Boolean	false	是否禁用
loading	Boolean	false	名称前是否带loading图标

续表

属 性 名	类　　型	默 认 值	说　　明
form-type	String		用于<form/>组件，点击分别会触发<form/>组件的submit/reset事件
open-type	String		微信开放能力
hover-class	String	button-hover	指定按钮按下去的样式类。当 hover-class="none" 时，没有点击态效果
hover-start-time	Number	20	按住后多久出现点击态，单位为毫秒
hover-stay-time	Number	70	手指松开后点击态保留时间，单位为毫秒

1．不同大小的按钮

按钮的大小有 default 和 mini 两种类型，如图 2.13 所示。

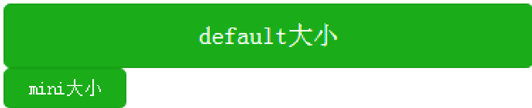


图 2.13　不同大小的按钮

实现代码如下所示。

```
<button size="default" type="primary">default 大小</button>
<button size="mini" type="primary">mini 大小</button>
```

2．不同类型的按钮

按钮的类型有 primary、default 和 warn 三种，如图 2.14 所示。

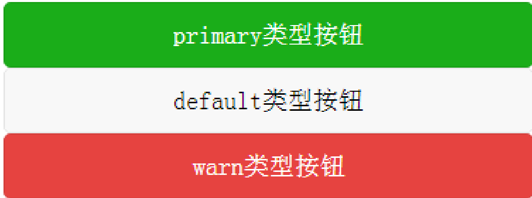


图 2.14　不同类型的按钮

实现代码如下所示。

```
<button size="default" type="primary">primary 类型按钮</button>
<button size="default" type="default">default 类型按钮</button>
<button size="default" type="warn">warn 类型按钮</button>
```

3．不同属性的按钮

按钮可以设置为镂空、禁用及名称前是否带 loading 图标，如图 2.15 所示。

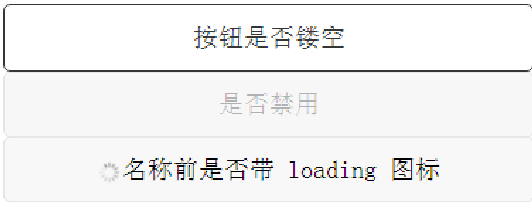


图 2.15 不同属性的按钮

实现代码如下所示。

```
<button size="default" plain>按钮是否镂空</button>
<button size="default" disabled>是否禁用</button>
<button size="default" loading>名称前是否带 loading 图标</button>
```

4．不同触发效果的按钮

按钮的触发效果分为两种：一种是 submit 提交表单，另一种是 reset 重置表单。reset 重置表单是通过 form-type 属性来设置的，需要配合 form 表单组件才能起作用，所有这个按钮触发效果会放在 form 表单组件来演示。

2.9 checkbox 多项选择器组件

checkbox 多项选择器组件是用来进行多项选择的组件，它的属性如表 2.7 所示。

表 2.7 多项选择器属性

属 性	类 型	默 认 值	说 明
value	String		<checkbox/>标识，选中时触发<checkbox-group/>的change事件，并携带<checkbox/>的value
disabled	Boolean	false	是否禁用
checked	Boolean	false	当前是否选中，可用来设置默认选中
color	Color		checkbox的颜色，同CSS的color

checkbox-group 可用来包裹多个 checkbox 多项选择器组件，可以绑定 bindchange 事件，<checkbox-group/>中选中项发生改变时会触发 change 事件，detail = {value:[选中的 checkbox 的 value 的数组]}。

在购物车的订单里，经常会用到 checkbox 多项选择器组件，如图 2.16 所示。



图 2.16　购物车

（1）在 app.js 全局文件里，先来初始化一些商品数据，并将其保存到本地缓存数据里，具体代码如下所示。

```
//app.js
App({
  onLaunch: function () {
    //调用 API 从本地缓存中获取数据
    var logs = wx.getStorageSync('logs') || []
    logs.unshift(Date.now())
    wx.setStorageSync('logs', logs);
    var goods = wx.getStorageSync('goods');
    if (!goods) {
      goods = this.loadGoods();
      wx.setStorageSync('goods', goods);
    }
  },
  getUserInfo: function (cb) {
    var that = this
    if (this.globalData.userInfo) {
```

```
        typeof cb == "function" && cb(this.globalData.userInfo)
      } else {
        //调用登录接口
        wx.login({
          success: function () {
            wx.getUserInfo({
              success: function (res) {
                that.globalData.userInfo = res.userInfo
                typeof cb == "function" && cb(that.globalData.userInfo)
              }
            })
          }
        })
      }
    },
    loadGoods: function () {
      var goods = new Array();
      var good = new Object();
      good.id = "0";
      good.pic = "/pages/images/supermarket/1-cz.jpg";
      good.name = "蒙牛纯甄酸牛奶原味 220g";
      good.price = "5.5";
      good.type = "milk.supermarket";
      good.count = "1";
      goods[0] = good;

      var good1 = new Object();
      good1.id = "1";
      good1.pic = "/pages/images/supermarket/1-hj.jpg";
      good1.name = "爱鲜蜂打火机 1 个";
      good1.price = "2";
      good1.type = "supermarket";
      good1.count = "1";
      goods[1] = good1;
      return goods;
    },
    globalData: {
      userInfo: null
    }
})
```

（2）在 checkbox.wxml 页面文件里，进行购物车页面的布局设计，具体代码如下所示。

```
<view class="content">
  <view class="info">
    <view class="sscs">
      <image src="/pages//images/cart/sscs.jpg" style="width:180px;height:47px;"></image>
    </view>
    <view class="line"></view>
    <view class="receive">
      <view>
        <view class="time">
          <view class="left">收货时间</view>
          <view class="right">一小时送达可预订</view>
        </view>
        <view class="freight">
          <image src="/pages//images/cart/shsj.jpg" style="width:24px;height:24px;"></image>
￥0 起送，22:00 前满￥30 免运费，22:00 后满￥69 免运费
     <view class="content">
  <view class="info">
    <view class="sscs">
      <image src="/pages//images/cart/sscs.jpg" style="width:180px;height:47px;"></image>
    </view>
    <view class="line"></view>
    <view class="receive">
      <view>
        <view class="time">
          <view class="left">收货时间</view>
          <view class="right">一小时送达可预订</view>
        </view>
        <view class="freight">
          <image src="/pages//images/cart/shsj.jpg" style="width:24px;height:24px;"></image>
￥0 起送，22:00 前满￥30 免运费，22:00 后满￥69 免运费
        </view>
      </view>
      <view class="detail2">></view>
    </view>
  </view>
  <view class="line"></view>
  <view class="items">
    <checkbox-group bindchange="checkboxChange">
```

```
        <block wx:for="{{orders}}">
          <view class="item">
            <view class="icon">
              <checkbox value="{{item.id}}" checked="{{selected}}" />
            </view>
            <view class="pic">
              <image src="{{item.pic}}" style="width:70px;height:87px;"></image>
            </view>
            <view class="order">
              <view class="title">{{item.name}}</view>
              <view class="priceInfo">
                <view class="price">￥{{item.price}}</view>
                <view class="minus" id="{{item.id}}" bindtap="minusGoods">-</view>
                <view class="count">{{item.count}}</view>
                <view class="add" id="{{item.id}}" bindtap="addGoods">+</view>
              </view>
            </view>
          </view>
          <view class="line"></view>
        </block>
      </checkbox-group>
      <checkbox-group bindchange="checkAll">
        <view class="all">
          <view>
            <checkbox checked="{{selectedAll}}" />
          </view>
          <view class="selectAll">
            全选
          </view>
          <view class="total">
            ￥{{totalPrice}}元
          </view>
          <view class="opr">
            选好了
          </view>
        </view>
      </checkbox-group>
  </view>
</view>
```

（3）在 checkbox.wxss 样式文件里，给购物车页面添加样式，具体代码如下所示。

```
.content{
    font-family: "Microsoft YaHei";
    height: 600px;
    background-color: #F9F9F8;
}

.info{
    background-color: #ffffff;
}
.sscs{
    text-align: center;
}
.line{
    border: 1px solid #cccccc;
    opacity: 0.2;
}
.receive{
    display: flex;
    flex-direction: row;
    padding:10px;
}
.time{
    display: flex;
    flex-direction: row;
}
.left{
    width: 50%;
    text-align: left;
    font-size: 15px;
    font-weight: bold;
}
.right{
    width:50%;
    text-align: right;
    font-size: 15px;
    color:#5DAFED;
    font-weight: bold;
    margin-right: 10px;
}
.freight{
```

```
    font-size: 13px;
    margin-right: 20px;
}
.detail2{
    position: absolute;
    right: 10px;
    line-height: 60px;
}
.items{
    background-color: #ffffff;
}
.item{
    display: flex;
    flex-direction: row;
    padding:10px;
    align-items: center;
}
.order{
    width: 100%;
    height: 87px;
}
.title{
    font-size: 15px;
}
.priceInfo{
    display: flex;
    flex-direction: row;
    margin-top:30px;
}
.price{
    width:65%;
    font-size: 15px;
    color: #ff0000;
    text-align: left;
}
.minus,.add{
    border: 1px solid #cccccc;
    border-radius: 25px;
    width: 25px;
    color: #ff0000;
```

```
        text-align: center;
}
.count{
        margin-left: 10px;
        margin-right: 10px;
}
.all{
        display: flex;
        flex-direction: row;
        height: 60px;
        align-items: center;
        padding-left: 10px;
}
.selectAll{
        width: 20%;
        text-align: center;
        font-size: 15px;
        font-weight: bold;
}
.total{
        width: 30%;
        font-size: 15px;
        color: #ff0000;
        font-weight: bold;
}
.opr{
        position: absolute;
        right: 0px;
        width: 92px;
        font-size: 15px;
        font-weight: bold;
        background-color: #FFD600;
        height: 60px;
        text-align: center;
        line-height: 60px;
}
```

（4）在 checkbox.js 业务逻辑处理文件里，给行内 checkbox 多项选择器组件和全局 checkbox 多项选择器组件添加函数，并初始化一些页面数据，具体代码如下所示。

```
Page({
    data: {
```

```
    orders: [],
    selected: true,
    selectedAll: true,
    totalPrice: 0
  },
  onLoad: function () {
    this.loadOrders();
  },
  loadOrders: function () {//加载购物车商品订单信息
    var orders = wx.getStorageSync('goods');
    this.setData({ orders: orders });
    var totalPrice = 0;
    for (var i = 0; i < orders.length; i++) {
      var order = orders[i];
      totalPrice += order.price * order.count;
    }
    this.setData({ totalPrice: totalPrice });
  },
  checkboxChange: function (e) {//商品行内 checkbox 多项选择器组件
    console.log(e);
    var ids = e.detail.value;
    if (ids.length == 0) {
      this.setData({ selectedAll: false });
    } else {
      this.setData({ selectedAll: true });
    }
    var orders = wx.getStorageSync('goods');
    var totalPrice = 0;
    for (var i = 0; i < orders.length; i++) {
      var order = orders[i];
      for (var j = 0; j < ids.length; j++) {
        if (order.id == ids[j]) {
          totalPrice += order.price * order.count
        }
      }
    }
    this.setData({ totalPrice: totalPrice });
  },
  checkAll: function (e) {//全选 checkbox 多项选择器组件
    var selected = this.data.selected;
```

```
        var result = selected == true ? false : true;
        this.setData({ selected: result });
        if (result == false) {
          this.setData({ totalPrice: 0 });
          this.setData({ selectedAll: false });
        } else {
          this.loadOrders();
          this.setData({ selectedAll: true });
        }

    }
})
```

这样就可以实现行内用 checkbox 多项选择器组件和全局 checkbox 多项选择器组件的函数来计算总金额，如图 2.17 和图 2.18 所示。



图 2.17　全选中



图 2.18　取消全选中

2.10　radio 单项选择器组件

radio 单项选择器每次只能选中一个选项，选项之间是一种互斥关系，是和 checkbox 多项选择器对立的一个组件，radio 单选选择器组件的属性如表 2.8 所示。

表 2.8　radio单项选择器组件的属性

属　　性	类　　型	默 认 值	说　　明
value	String		<radio/> 标识。当该<radio/> 选中时，<radio-group/> 的 change 事件会携带<radio/>的value
disabled	Boolean	false	是否禁用
checked	Boolean	false	当前是否选中，可用来设置默认选中
color	Color		radio的颜色，同CSS的color

radio-group 可用来包裹多个 radio 单项选择器组件，它有一个绑定事件 bindchange，<radio-group/> 中的选中项发生变化时触发 change 事件，event.detail = {value: 选中项 radio 的 value}。

radio 单项选择器经常用来进行性别的选择，性别只能二选一：男或者女。

（1）在 radio.wxml 页面文件里，使用 radio-group 和 radio 单项选择器进行页面的布局，具体代码如下所示。

```
<radio-group class="radio-group" bindchange="radioChange">
  <label class="radio" wx:for="{{items}}">
    <radio value="{{item.name}}" checked="{{item.checked}}"/>{{item.value}}
  </label>
</radio-group>
```

（2）在 radio.js 业务逻辑处理文件里，初始化性别数据以及 radioChange 绑定的函数，具体代码如下所示。

```
Page({
  data: {
    items: [
      { name: 'man', value: '男', checked: 'true'},
      { name: 'women', value: '女' }
    ]
  },
  radioChange: function (e) {
    console.log('radio 发生 change 事件，携带 value 值为：', e.detail.value)
  }
})
```

2.11　input 单行输入框组件

input 单行输入框组件是用来输入单行文本的组件，它的属性如表 2.9 所示。

表 2.9　单行输入框的属性

属　性	类　型	默 认 值	说　明
value	String		输入框的初始内容
type	String	text	input的类型，有效值：text文本输入键盘，number数字输入键盘，idcard身份证输入键盘，digit带小数点的数字键盘
password	Boolean	false	是否是密码类型
placeholder	String		输入框为空时占位符
placeholder-style	String		指定 placeholder 的样式
placeholder-class	String	input-placeholder	指定 placeholder 的样式类
disabled	Boolean	false	是否禁用
maxlength	Number	140	最大输入长度，设置为 -1的时候不限制最大长度
cursor-spacing	Number	0	指定光标与键盘的距离，单位px。取input距离底部的距离和cursor-spacing指定的距离的最小值作为光标与键盘的距离
auto-focus	Boolean	false	（即将废弃，请直接使用focus）自动聚焦，拉起键盘
focus	Boolean	false	获取焦点
confirm-type	String	done	设置键盘右下角按钮的文字：send（发送），search（搜索），next（下一个），go（前往），done（完成）
confirm-hold	Boolean	false	单击键盘右下角按钮时是否保持键盘不收起
bindinput	EventHandle		当键盘输入时，触发input事件，event.detail = {value: value}，处理函数可以直接 return 一个字符串，替换输入框的内容
bindfocus	EventHandle		输入框聚焦时触发，event.detail = {value: value}
bindblur	EventHandle		输入框失去焦点时触发，event.detail = {value: value}
bindconfirm	EventHandle		单击完成按钮时触发，event.detail = {value: value}

input 单行输入框一般通过属性来设置，比如设置输入框的默认值、设置输入框为密码类型、设置输入框的占位符提示、设置输入框的占位符样式、设置输入框禁用以及设置输

入框最大长度，如图 2.19 所示。

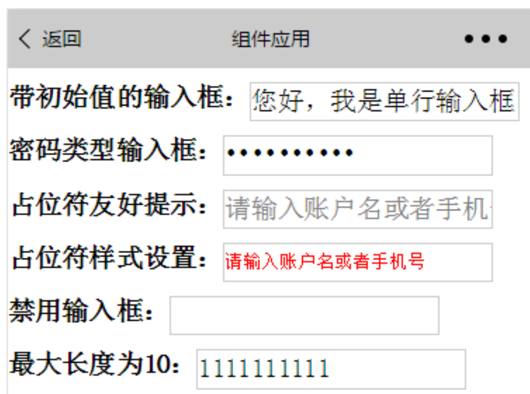


图 2.19 输入框属性设置

实现代码如下所示。

```
<view style="display:flex;flex-direction:row;margin-top:10px;">
   <view style="font-weight:bold;">带初始值的输入框：</view>
   <view><input type="text" value="您好，我是单行输入框" style="border:1px solid #cccccc" /><
/view>
</view>

<view style="display:flex;flex-direction:row;margin-top:10px;">
   <view style="font-weight:bold;">密码类型输入框：</view>
   <view><input type="text" password value="您好，我是单行输入框"  style="border:1px solid #
cccccc"/></view>
</view>

<view style="display:flex;flex-direction:row;margin-top:10px;">
   <view style="font-weight:bold;">占位符友好提示：</view>
   <view><input type="text" placeholder="请输入账户名或者手机号" style="border:1px solid #cc
cccc"/></view>
</view>

<view style="display:flex;flex-direction:row;margin-top:10px;">
   <view style="font-weight:bold;">占位符样式设置：</view>
   <view><input type="text" placeholder="请输入账户名或者手机号" placeholder-style="font-size:
13px;color:red;" style="border:1px solid #cccccc"/></view>
</view>
```

```
<view style="display:flex;flex-direction:row;margin-top:10px;">
  <view style="font-weight:bold;">禁用输入框：</view>
  <view><input type="text" disabled style="border:1px solid #cccccc"/></view>
</view>

<view style="display:flex;flex-direction:row;margin-top:10px;">
  <view style="font-weight:bold;">最大长度为 10：</view>
  <view><input type="text" maxlength="10" style="border:1px solid #cccccc"/></view>
</view>
```

2.12　textarea 多行输入框组件

textarea 多行输入框组件是用来输入多行文本的组件，当需要输入多行文本时，就可以选择这个组件，它的属性和 input 单行输入框组件的属性类似，textarea 多行输入框组件的属性如表 2.10 所示。

表 2.10　textarea多行输入框组件的属性

属　　性	类　　型	默 认 值	说　　明
value	String		输入框的内容
placeholder	String		输入框为空时占位符
placeholder-style	String		指定 placeholder 的样式
placeholder-class	String	input-placeholder	指定 placeholder 的样式类
disabled	Boolean	false	是否禁用
maxlength	Number	140	最大输入长度，设置为 -1 的时候不限制最大长度
cursor-spacing	Number	0	指定光标与键盘的距离，单位 px 。取 textarea 距离底部的距离和 cursor-spacing 指定的距离的最小值作为光标与键盘的距离
auto-focus	Boolean	false	（即将废弃，请直接使用 focus ）自动聚焦，收起键盘
focus	Boolean	false	获取焦点

多行文本输入的场景使用还是比较多的，就像很多 App 软件都有反馈填写的功能。在填写反馈信息的时候，需要输入多行文本内容，这时就可以使用 textarea 多行输入框组件，如图 2.20 所示。

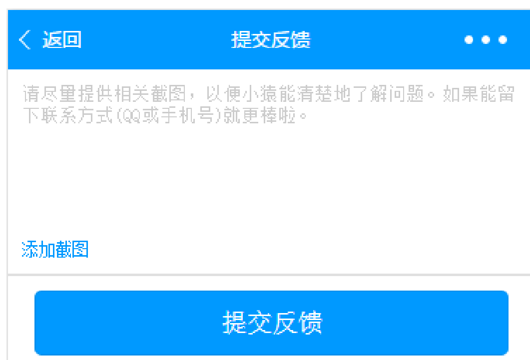


图 2.20 猿题库问题反馈

（1）在 textarea.json 配置文件里来配置窗口的表现，导航标题为“提交反馈”，白色（#FFFFFF）字体，背景颜色为蓝色（#0099FF），具体代码如下所示。

```
{
  "backgroundTextStyle": "light",
  "navigationBarBackgroundColor": "#0099FF",
  "navigationBarTitleText": "提交反馈",
  "navigationBarTextStyle": "white"
}
```

（2）在 textarea.wxml 页面文件里进行反馈界面的布局，提供可以输入多行文本的区域以及提交按钮，具体代码如下所示。

```
<form bindsubmit="bindFormSubmit">
<view style="margin:10px;">
   <textarea placeholder="请尽量提供相关截图，以便小猿能清楚地了解问题。如果能留下联系方式（QQ 或手机号）就更棒啦。" placeholder-style="font-size:13px;color:#cccccc;" style="width:100%;height:100px;" name="textarea"/>
</view>
<view style="color:#0099FF;font-size:12px;margin-left:10px;margin-bottom:10px;">添加截图</view>
<view style="border:1px solid #666666;opacity:0.2;"></view>
<button style="width:90%;background:#0099FF;color:#FFFFFF;margin-top:10px;" form-type="submit">提交反馈</button>
</form>
```

（3）在 textarea.js 业务逻辑处理文件里，通过 textara 多行输入框组件的 name 属性来获取输入的值，具体代码如下所示。

```
Page({
  bindFormSubmit: function (e) {
```

```
        console.log(e.detail.value.textarea)
    }
})
```

2.13　label 改进表单组件

label 改进表单组件可用来改进表单可用性，目前可以用来绑定和改进的组件有<button />、<checkbox/>、<radio/>和 <switch/>这四种组件，label 组件只有一个 for 属性，使用 for 属性可以找到绑定控件的 id。

它的使用方式有两种：一种是定义了 for 属性，另一种是没有定义 for 属性。

1．label 组件没有定义 for 属性

label 改进表单组件没有定义 for 属性时，如果在 label 组件内包含<button/>、<checkbox/>、<radio/>、<switch/>这些组件，则示例代码如下所示。

```
<label>
    <button  bindtap="clickBtn"  hidden>保存</button>
    <view>我是 label 组件内的内容</view>
    <checkbox-group  bindchange="checkboxChange">
        <checkbox  value="音乐"  />音乐
        <checkbox  value="电商"  />电商
    </checkbox-group>
    <radio-group  bindchange="radioChange">
        <radio  value="白天"/>白天
        <radio  value="晚上"/>晚上
    </radio-group>
</label>
```

从示例代码里可以看出，label 组件内包含 button 组件、view 组件、checkbox 组件和 radio 组件这四种。当单击 label 组件时，会触发 label 组件内包含的第一个组件，也就是 button 所提供的点击事件。假如 checkbox 组件在第一位，就会触发 checkbox 对应的事件。

下面看看没有使用 label 改进表单之前和改进表单之后的区别，如图 2.21 和图 2.22 所示。



图 2.21　没有改进表单

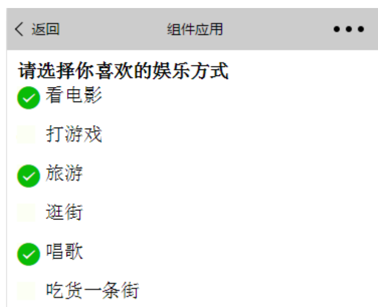


图 2.22　改进表单

（1）在 label.wxml 页面文件里进行页面的布局，来设计娱乐方式的多选，具体代码如下所示。

```
<view class="section section_gap">
  <view class="section__title">请选择你喜欢的娱乐方式</view>
  <checkbox-group class="group" bindchange="checkboxChange">
    <view class="label-1" wx:for="{{checkboxItems}}">
      <label>
        <checkbox  value="{{item.name}}" checked="{{item.checked}}"></checkbox>
        <view class="label-1__icon">
            <icon hidden type="success" style="opacity:{{item.checked ? 1: 0}}"/>
        </view>
        <text class="label-1__text">{{item.value}}</text>
      </label>
    </view>
  </checkbox-group>
</view>

<view class="section section_gap">
  <view class="section__title">请选择你喜欢的娱乐方式</view>
  <checkbox-group class="group" bindchange="checkboxChange">
    <view class="label-1" wx:for="{{checkboxItems}}">
      <label>
        <checkbox hidden value="{{item.name}}" checked="{{item.checked}}"></checkbox>
        <view class="label-1__icon">
            <icon  type="success" style="opacity:{{item.checked ? 1: 0}}"/>
        </view>
        <text class="label-1__text">{{item.value}}</text>
      </label>
```

```
    </view>
  </checkbox-group>
</view>
```

（2）在 label.wxss 样式文件里，给页面布局添加样式，具体代码如下所示。

```
.section{
  margin:10px;
}
.section__title{
  font-weight: bold;
}
.label-1, .label-2{
    margin-bottom: 15px;
}
.label-1__text, .label-2__text {
    display: inline-block;
    vertical-align: middle;
}

.label-1__icon {
    position: relative;
    margin-right: 10px;
    display: inline-block;
    vertical-align: middle;
    width: 18px;
    height: 18px;
    background: #fcfff4;
}

.label-1__icon-checked {
    position: absolute;
    top: 3px;
    left: 3px;
    width: 12px;
    height: 12px;
    background: #1aad19;
}


.label-2__icon {
    position: relative;
```

```
    display: inline-block;
    vertical-align: middle;
    margin-right: 10px;
    width: 18px;
    height: 18px;
    background: #fcfff4;
    border-radius: 50px;
}

.label-2__icon-checked {
    position: absolute;
    left: 3px;
    top: 3px;
    width: 12px;
    height: 12px;
    background: #1aad19;
    border-radius: 50%;
}

.label-4_text{
    text-align: center;
    margin-top: 15px;
}
```

（3）在 label.js 业务逻辑处理页面里，初始化 checkbox 多项选择器数据以及相应的触发事件，具体代码如下所示。

```
Page({
  data: {
    checkboxItems: [
      { name: 'kdy', value: '看电影' },
      { name: 'dyx', value: '打游戏'},
      { name: 'ly', value: '旅游' },
      { name: 'gj', value: '逛街' },
      { name: 'cg', value: '唱歌' },
      { name: 'ch', value: '吃货一条街' },
    ],
    hidden: false
  },
  checkboxChange: function (e) {
    var checked = e.detail.value
    var changed = {}
```

```
    for (var i = 0; i < this.data.checkboxItems.length; i++) {
      if (checked.indexOf(this.data.checkboxItems[i].name) !== -1) {
        changed['checkboxItems[' + i + '].checked'] = true
      } else {
        changed['checkboxItems[' + i + '].checked'] = false
      }
    }
    this.setData(changed)
  }
})
```

从这里可以看出，当 label 组件内有多个组件时，会触发第一个组件。

2．label 组件定义了 for 属性

微信小程序 label 组件定义了 for 属性后，会根据 for 属性的值找到对应的组件 id，然后触发这个组件的相应事件，label 组件同样可以实现改进表单的效果，如图 2.23 和图 2.24 所示。



图 2.23　改进表单前



图 2.24　改进表单后

（1）在 label.wxml 页面文件里进行页面的布局，设计薪水范围的调查，具体代码如下所示。

```
<view class="section section_gap">
  <view class="section__title">调查你的薪水范围</view>
  <radio-group class="group" bindchange="radioChange">
    <view class="label-2" wx:for="{{radioItems}}">
      <radio id="{{item.name}}" value="{{item.name}}" checked="{{item.checked}}"></radio>
      <view class="label-2__icon">
        <icon hidden type="success" style="opacity:{{item.checked ? 1: 0}}"/>
      </view>
      <label class="label-2__text" for="{{item.name}}">
```

```
            <text>{{item.value}}</text>
         </label>
      </view>
   </radio-group>
</view>

<view  class="section  section_gap">
   <view  class="section__title">调查你的薪水范围</view>
   <radio-group  class="group"  bindchange="radioChange">
      <view  class="label-2"  wx:for="{{radioItems}}">
         <radio  id="{{item.name}}"  hidden  value="{{item.name}}"  checked="{{item.checked}}"></ra
dio>
         <view  class="label-2__icon">
            <icon  type="success"  style="opacity:{{item.checked  ?  1:  0}}"/>
         </view>
         <label  class="label-2__text"  for="{{item.name}}">
            <text>{{item.value}}</text>
         </label>
      </view>
   </radio-group>
</view>
```

（2）在 label.wxss 样式文件里，给页面布局添加样式，具体代码如下所示。

```
.section{
   margin:10px;
}
.section__title{
   font-weight: bold;
}
.label-1, .label-2{
      margin-bottom: 15px;
}
.label-1__text, .label-2__text {
      display: inline-block;
      vertical-align: middle;
}

.label-1__icon {
      position: relative;
      margin-right: 10px;
      display: inline-block;
```

```
    vertical-align: middle;
    width: 18px;
    height: 18px;
    background: #fcfff4;
}

.label-1__icon-checked {
    position: absolute;
    top: 3px;
    left: 3px;
    width: 12px;
    height: 12px;
    background: #1aad19;
}


.label-2__icon {
    position: relative;
    display: inline-block;
    vertical-align: middle;
    margin-right: 10px;
    width: 18px;
    height: 18px;
    background: #fcfff4;
    border-radius: 50px;
}

.label-2__icon-checked {
    position: absolute;
    left: 3px;
    top: 3px;
    width: 12px;
    height: 12px;
    background: #1aad19;
    border-radius: 50%;
}

.label-4_text{
    text-align: center;
    margin-top: 15px;
```

```
}
```

（3）在 label.js 业务逻辑处理页面里，初始化 radio 单项选择器数据以及相应的触发事件，具体代码如下所示。

```
Page({
  data: {
    radioItems: [
      { name: '1', value: '1000-2000 元' },
      { name: '2', value: '2000-4000 元', checked: 'true' },
      { name: '3', value: '4000-6000 元' },
      { name: '4', value: '6000-8000 元' },
      { name: '5', value: '8000-10000 元' },
      { name: '6', value: '10000-20000 元' },
    ],
    hidden: false
  },
    radioChange: function (e) {
    var checked = e.detail.value
    var changed = {}
    for (var i = 0; i < this.data.radioItems.length; i++) {
      if (checked.indexOf(this.data.radioItems[i].name) !== -1) {
        changed['radioItems[' + i + '].checked'] = true
      } else {
        changed['radioItems[' + i + '].checked'] = false
      }
    }
    this.setData(changed)
  }
})
```

从这里可以看出，如果 label 定义了 for 属性，那么它就会根据 for 属性的值找到对应的组件 id，然后触发相应事件。如果 label 没有定义 for 属性，那么它会找到 label 组件内的第一个组件，然后触发相应事件。

2.14 picker 选择器组件

picker 选择器组件提供了三种类型的选择器，分别为普通选择器、时间选择器和日期选择器。普通选择器用来显示一个数组，比如城市地区数组、学历数组等内容，都可以用

普通选择器。时间选择器用来选择时和秒的时间。日期选择器就是用来选择一个日期。通过 picker 选择器组件的 mode 属性可以区分到底是哪一种选择器，默认是普通选择器。

2.14.1　picker 普通选择器组件

picker 选择器默认的是普通选择器，mode 属性等于 selector，它的属性如表 2.11 所示。

表 2.11　普通选择器组件的属性

属　　性	类　　型	默认值	说　　明
range	Array / Object Array	[]	mode为selector时，range有效
range-key	String		当range是一个Object Array时，通过range-key来指定Object中key的值作为选择器显示内容
value	Numbe	0	value的值表示选择了range中的第几个（下标从0开始）
bindchange	EventHandle		value改变时触发change事件，event.detail = {value: value}
disabled	Boolean	false	是否禁用

当想从一些内容里面选择一项的时候就可以使用普通选择器，比如选择学历，即从中专、大专、本科、硕士、博士五种学历中选择一种，如图 2.25 所示。



图 2.25　学历选择器

（1）在 picker.wxml 页面文件里，使用 picker 选择器进行界面布局，具体代码如下所示。

```
<view class="section">
```

```
  <view class="section__title">学历选择器</view>
  <picker bindchange="bindPickerChange" value="{{index}}" range="{{array}}">
    <view class="picker">
      当前选择：{{array[index]}}
    </view>
  </picker>
</view>
```

（2）在 picker.js 业务逻辑处理文件里，初始化学历选择器的数据以及选择绑定的事件，具体代码如下所示。

```
Page({
  data: {
    array: ['中专', '大专', '本科', '硕士','博士'],
    objectArray: [
      {
        id: 0,
        name: '中专'
      },
      {
        id: 1,
        name: '大专'
      },
      {
        id: 2,
        name: '本科'
      },
      {
        id: 3,
        name: '硕士'
      },
      {
        id: 4,
        name: '博士'
      }
    ],
    index: 0
  },
  bindPickerChange: function (e) {
    console.log('picker 发送选择改变，携带值为', e.detail.value)
    this.setData({
      index: e.detail.value
```

```
    })
  }
})
```

2.14.2　picker 时间选择器组件

当 picker 选择器的 mode 属性等于 time 时，就是时间选择器，它的属性如表 2.12 所示。

表 2.12　时间选择器组件的属性

属　　性	类　　型	默认值	说　　明
value	String		表示选中的时间，格式为"hh:mm"
start	String		表示有效时间范围的开始，字符串格式为"hh:mm"
end	String		表示有效时间范围的结束，字符串格式为"hh:mm"
bindchange	EventHandle		value改变时触发change事件，event.detail = {value: value}
disabled	Boolean	false	是否禁用

picker 时间选择器的格式是“hh:mm”，它在起始时间和结束时间之间选择一个时间，如图 2.26 所示。



图 2.26　时间选择器

（1）在 picker.wxml 页面文件里，使用 picker 选择器进行界面布局，具体代码如下所示。

```
<view  class="section">
```

```
  <view class="section__title">时间选择器</view>
  <picker mode="time" value="{{time}}" start="09:00" end="23:59" bindchange="bindTimeChan
ge">
    <view class="picker">
      当前选择: {{time}}
    </view>
  </picker>
</view>
```

（2）在 picker.js 业务逻辑处理文件里，初始化时间选择器的数据并选择绑定的事件，具体代码如下所示。

```
Page({
  data: {
    time: '21:01'
  },
  bindTimeChange: function (e) {
    this.setData({
      time: e.detail.value
    })
  }
})
```

2.14.3 picker 日期选择器组件

picker 选择器的 mode 属性等于 date 时，就是日期选择器，它的属性如表 2.13 所示。

表 2.13 日期选择器组的属性

属　性	类　型	默认值	说　明
value	String	0	表示选中的日期，格式为“YYYY-MM-DD”
start	String		表示有效日期范围的开始，字符串格式为“YYYY-MM-DD”
end	String		表示有效日期范围的结束，字符串格式为“YYYY-MM-DD”
fields	String	day	有效值 year、month、day，表示选择器的粒度
bindchange	EventHandle		value改变时触发change事件，event.detail = {value: value}
disabled	Boolean	false	是否禁用

picker 时间选择器的格式是“YYYY-MM-DD”，它在起始日期和结束日期之间选择一个日期，如图 2.27 所示。



图 2.27　日期选择器

（1）在 picker.wxml 页面文件里，使用 picker 选择器进行界面布局，具体代码如下所示。

```
<view class="section">
  <view class="section__title">日期选择器</view>
  <picker mode="date" value="{{date}}" start="2017-01-01" end="2017-12-01" bindchange="bindDateChange">
    <view class="picker">
      当前选择: {{date}}
    </view>
  </picker>
</view>
```

（2）在 picker.js 业务逻辑处理文件里，初始化日期选择器的数据并选择绑定的事件，具体代码如下所示。

```
Page({
  data: {
    date: '2017-06-10'
  },
  bindDateChange: function (e) {
    this.setData({
      date: e.detail.value
    })
  }
})
```

2.15　picker-view 嵌入页面的滚动选择器

在 picker-view 组件内只能放置 picker-view-column 组件，其他节点不会显示。picker-view-column 的高度会自动设置成与 picker-view 的选中框的高度一致。具体属性如表 2.14 所示。

表 2.14　picker-view嵌入页面的滚动选择器组件的属性

属　　性	类　　型	默认值	说　　明
value	Number	Array	数组中的数字依次表示picker-view内的picker-view-colume选择的第几项（下标从0开始），数字大于picker-view-column 可选项长度时，选择最后一项
indicator-style	String		设置选择器中间选中框的样式
indicator-class	String		设置选择器中间选中框的类名
bindchange	EventHandle		当滚动时选择，value 改变时触发 change 事件，event.detail = {value: value}；value为数组，表示 picker-view 内的 picker-view-column 当前选择的是第几项（下标从 0 开始）

picker 选择器是以弹出层的形式来展现选择器，而 picker-view 组件是一款嵌入页面的滚动选择器，可以直接来选择日期，如图 2.28 所示。



图 2.28　picker-view 嵌入页面的滚动选择器

（1）在 picker-view.wxml 文件内进行 picker-view 嵌入页面的滚动选择器布局，具体代码如下所示。

```
<view>
  <view>{{year}}年{{month}}月{{day}}日</view>
  <picker-view    indicator-style="height:    50px;"    style="width:    100%;    height:    300px;"
```

```
value="{{value}}" bindchange="bindChange">
    <picker-view-column>
      <view wx:for="{{years}}" style="line-height: 50px">{{item}}年</view>
    </picker-view-column>
    <picker-view-column>
      <view wx:for="{{months}}" style="line-height: 50px">{{item}}月</view>
    </picker-view-column>
    <picker-view-column>
      <view wx:for="{{days}}" style="line-height: 50px">{{item}}日</view>
    </picker-view-column>
  </picker-view>
</view>
```

（2）在 picker-view.js 业务逻辑处理文件里，初始化嵌入页面的滚动选择器的数据并进行相应的事件处理，具体代码如下所示。

```
const date = new Date()
const years = []
const months = []
const days = []

for (let i = 1990; i <= date.getFullYear(); i++) {
  years.push(i)
}

for (let i = 1; i <= 12; i++) {
  months.push(i)
}

for (let i = 1; i <= 31; i++) {
  days.push(i)
}

Page({
  data: {
    years: years,
    year: date.getFullYear(),
    months: months,
    month: 2,
    days: days,
    day: 2,
    year: date.getFullYear(),
```

```
        value: [9999, 1, 1],
    },
    bindChange: function (e) {
        const val = e.detail.value
        this.setData({
            year: this.data.years[val[0]],
            month: this.data.months[val[1]],
            day: this.data.days[val[2]]
        })
    }
})
```

2.16　slider 滑动选择器组件

slider 滑动选择器组件经常被用来设置屏幕的亮度、声音的大小等，具体属性如表 2.15 所示。

表 2.15　slider滑动选择器组件的属性

属　　性	类　　型	默认值	说　　明
min	Number	0	最小值
max	Number	100	最大值
step	Number	1	步长，取值必须大于 0，并且可被(max - min)整除
disabled	Boolean	false	是否禁用
value	Number	0	当前取值
color	Color	#e9e9e9	背景条的颜色
selected-color	Color	#1aad19	已选择的颜色
show-value	Boolean	false	是否显示当前 value
bindchange	EventHandle		完成一次拖动后触发的事件，event.detail = {value: value}

slider 滑动选择器组件可以设置最小值、最大值、步长、是否禁用、当前值、背景条颜色、已选择的颜色、是否显示当前值等属性的使用，如图 2.29 所示。

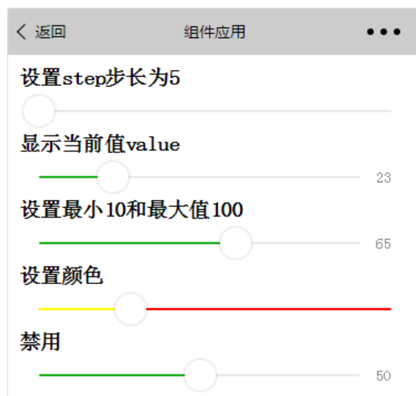


图 2.29　slider 滑动选择器的属性

示例代码如下所示。

```
<view class="section section_gap" style="margin:10px;">
  <text class="section__title" style="font-weight:bold;">设置 step 步长为 5</text>
  <view class="body-view">
    <slider bindchange="sliderchange" step="5"/>
  </view>
</view>

<view class="section section_gap" style="margin:10px;">
  <text class="section__title" style="font-weight:bold;">显示当前值 value</text>
  <view class="body-view">
    <slider bindchange="sliderchange" show-value value="23"/>
  </view>
</view>

<view class="section section_gap" style="margin:10px;">
  <text class="section__title" style="font-weight:bold;">设置最小 10 和最大值 100</text>
  <view class="body-view">
    <slider bindchange="sliderchange" min="10" max="100" show-value/>
  </view>
</view>

<view class="section section_gap" style="margin:10px;">
  <text class="section__title" style="font-weight:bold;">设置颜色</text>
  <view class="body-view">
    <slider bindchange="sliderchange"  color="red" selected-color="yellow"/>
```

```
    </view>
</view>

<view class="section section_gap" style="margin:10px;">
    <text class="section__title" style="font-weight:bold;">禁用</text>
    <view class="body-view">
        <slider bindchange="sliderchange" disabled show-value value="50"/>
    </view>
</view>
```

2.17 switch 开关选择器组件

switch 开关选择器组件可用来控制状态的变更，比如声音的状态可以是静音或震动、人的状态可以是好人或坏人，这样状态的设置就可以通过 switch 开关选择器组件来控制，它的属性如表 2.16 所示。

表 2.16 switch开关选择器组件的属性

属　　性	类　　型	默 认 值	说　　明
checked	Boolean	false	是否选中
type	String	switch	样式，有效值：switch, checkbox
bindchange	EventHandle		checked 改变时触发 change 事件，event.detail={ value:checked}
color	Color		switch 的颜色，同CSS的color

switch 开关选择器的属性可以设置为是否选中、开关类型、颜色以及绑定事件，利用这些属性就可以设置状态的变化。在猿题库 App 的科目设置界面就使用了 switch 开关选择器来动态控制科目的显示，如图 2.30 所示。



图 2.30　猿题库科目设置

（1）在 switch.wxml 页面文件里，进行猿题库科目设置界面的布局，具体代码如下所示。

```
<view class="content">
<block wx:for="{{subjects}}">
<view class="item">
    <view class="pic"><image src="{{item.pic}}" style="width:28px;height:33px;"></image></view>
    <view class="subject">
        <view class="name">{{item.name}}</view>
        <view class="desc">{{item.desc}}</view>
    </view>
    <view class="btn">
      <switch checked="{{item.status == '1'?true:''}}" type="switch" bindchange="switchChange" id="{{item.id}}"/>
    </view>
</view>
<view class="line"></view>
</block>
</view>
```

（2）在 switch.wxss 样式文件里，给猿题库科目设置界面的添加样式，具体代码如下所示。

```
.item{
    display: flex;
    flex-direction: row;
    padding:10px;
}
.pic{
    width: 10%;
}
.subject{
    width: 75%;
}
.btn{
    width:15%;
}
.name{
    font-weight: bold;
}
.desc{
    font-size: 12px;
    margin-top:10px;
    color: #666666;
}
.line{
    border: 1px dotted #cccccc;
}
```

（3）在 switch.js 业务逻辑处理文件里，初始化猿题库的科目并添加 switchChange 开关改变触发事件，具体代码如下所示。

```
Page({
  data: {
    subjects: []
  },
  onLoad: function (options) {
    var subjects = this.loadSubjects();
    wx.setStorageSync("subjects", subjects);
    this.setData({ subjects: wx.getStorageSync("subjects")});
  },
  switchChange: function (e) {
```

```
    var status = e.detail.value;
    var id = e.target.id;
    var subjects = wx.getStorageSync("subjects");
    var result = new Array();
    for (var i = 0; i < subjects.length; i++) {
      if (subjects[i].id == id) {
        subjects[i].status = status == true ? '1' : '2';
      }
      result.push(subjects[i]);
    }
    wx.setStorageSync('subjects', result);
    this.setData({ subjects: result });
  },
  loadSubjects: function () {
    var subjects = new Array();

    var subject = new Object();
    subject.id = "1";
    subject.pic = "/pages/images/subject/yuwen.jpg";
    subject.name = "语文";
    subject.status = "1";
    subject.desc = "支持初中语文所有教材版本的同步练习，覆盖中考考纲的冲刺总复习";
    subjects.push(subject);

    var subject1 = new Object();
    subject1.id = "2";
    subject1.pic = "/pages/images/subject/shuxue.jpg";
    subject1.name = "数学";
    subject1.status = "1";
    subject1.desc = "支持初中数学所有教材版本的同步练习，覆盖中考考纲的冲刺总复习";
    subjects.push(subject1);

    var subject2 = new Object();
    subject2.id = "3";
    subject2.pic = "/pages/images/subject/yingyu.jpg";
    subject2.name = "英语";
    subject2.status = "1";
    subject2.desc = "支持初中英语所有教材版本的同步练习，覆盖中考考纲的冲刺总复习";
    subjects.push(subject2);
```

```
var subject3 = new Object();
subject3.id = "4";
subject3.pic = "/pages/images/subject/wuli.jpg";
subject3.name = "物理";
subject3.status = "1";
subject3.desc = "支持初中物理所有教材版本的同步练习，覆盖中考考纲的冲刺总复习";
subjects.push(subject3);

var subject4 = new Object();
subject4.id = "5";
subject4.pic = "/pages/images/subject/huaxue.jpg";
subject4.name = "化学";
subject4.status = "1";
subject4.desc = "支持初中化学所有教材版本的同步练习，覆盖中考考纲的冲刺总复习";
subjects.push(subject4);

var subject5 = new Object();
subject5.id = "6";
subject5.pic = "/pages/images/subject/shengwu.jpg";
subject5.name = "生物";
subject5.status = "1";
subject5.desc = "支持初中生物所有教材版本的同步练习，覆盖中考考纲的冲刺总复习";
subjects.push(subject5);

var subject6 = new Object();
subject6.id = "7";
subject6.pic = "/pages/images/subject/lishi.jpg";
subject6.name = "历史";
subject6.status = "1";
subject6.desc = "支持初中历史所有教材版本的同步练习，覆盖中考考纲的冲刺总复习";
subjects.push(subject6);

var subject7 = new Object();
subject7.id = "8";
subject7.pic = "/pages/images/subject/dili.jpg";
subject7.name = "地理";
subject7.status = "2";
subject7.desc = "支持初中地理所有教材版本的同步练习，覆盖中考考纲的冲刺总复习";
subjects.push(subject7);
```

```
        var subject8 = new Object();
        subject8.id = "9";
        subject8.pic = "/pages/images/subject/kexue.jpg";
        subject8.name = "科学";
        subject8.status = "2";
        subject8.desc = "支持初中科学所有教材版本的同步练习，覆盖中考考纲的冲刺总复习";
        subjects.push(subject8);

        var subject9 = new Object();
        subject9.id = "10";
        subject9.pic = "/pages/images/subject/shehui.jpg";
        subject9.name = "历史与社会";
        subject9.status = "2";
        subject9.desc = "支持初中历史与社会所有教材版本的同步练习，覆盖中考考纲的冲刺总复习";
        subjects.push(subject9);

        var subject10 = new Object();
        subject10.id = "11";
        subject10.pic = "/pages/images/subject/sixiang.jpg";
        subject10.name = "思想品德";
        subject10.status = "2";
        subject10.desc = "支持初中思想品德所有教材版本的同步练习，覆盖中考考纲的冲刺总复习";
        subjects.push(subject10);

        return subjects;
    }
})
```

2.18　form 表单组件

form 表单组件把表单里<switch/>、<input/>、<checkbox/>、<slider/>、<radio/>和<picker/>这些组件的值提交给 js 进行处理，提交表单的时候，会借助于 button 组件的 formType 为 submit 的属性，提交表单组件中的 value 值，并且需要在表单组件中加上 name 作为 key。form 表单组件的属性如表 2.17 所示。

表 2.17　form表单组件的属性

属　　性	类　　型	默认值	说　　明
report-submit	Boolean		是否返回 formId，用于发送模板消息
bindsubmit	EventHandle		携带 form 中的数据触发 submit 事件，event.detail = {value : {'name': 'value'} , formId: ''}
bindreset	EventHandle		表单重置时会触发 reset 事件

表单是 App 软件和网站不可或缺的一部分，为了获取用户的一些信息，比如婚博会的索票表单，就是先让你填写一些个人资料后，才会发放门票给你，如图 2.31 和图 2.32 所示。



图 2.31　免费索票



图 2.32　获知渠道

（1）在 form.wxml 页面文件里，进行婚博会免费索票的界面布局和获知渠道的界面布局，具体代码如下所示。

```
<view class="content">
<form bindsubmit="formSubmit">
        <view class="item">
            <view class="name">姓名：</view>
            <view class="val"><input name="name" type="text"/></view>
        </view>
        <view class="item">
            <view class="name">手机号：</view>
            <view class="val"><input name="mobile" type="text"/></view>
        </view>
```

```
        <view class="item">
            <view class="name">爱人姓名：</view>
            <view class="val"><input name="lovename" type="text"/></view>
        </view>
        <view class="item">
            <view class="name">爱人手机：</view>
            <view class="val"><input name="lovemobile" type="text"/></view>
        </view>
        <view class="item">
            <view class="name">快递地址：</view>
            <view class="val"><input name="address" type="text" placeholder="情侣套票免费快递
到家" placeholder-class="holder"/></view>
        </view>
        <view class="item">
            <view class="name">婚期：</view>
            <view class="val"><input name="date" type="text" placeholder="请输入日期"
placeholder-class="holder"/></view>
        </view>
        <view class="item">
            <view class="name">需要筹备：</view>
            <view class="box">
               <checkbox-group name="box">
                   <checkbox value="订喜宴">订喜宴</checkbox>
                   <checkbox value="找婚庆">找婚庆</checkbox>
                   <checkbox value="拍婚照">拍婚照</checkbox>
                   <checkbox value="订婚戒">订婚戒</checkbox>
                   <checkbox value="选婚纱">选婚纱</checkbox>
                   <checkbox value="度蜜月">度蜜月</checkbox>
                   <checkbox value="淘婚品">淘婚品</checkbox>
                   <checkbox value="美新娘">美新娘</checkbox>
                   <checkbox value="租婚车">租婚车</checkbox>
               </checkbox-group>
            </view>
        </view>
         <view class="item">
            <view class="name">获知渠道：</view>
            <view><button class="way" bindtap="selectWay">{{way}}</button></view>
        </view>
        <button class="btn" form-type="submit">免费索票</button>
    </form>
```

```
</view>
<view class="{{flag=='0'?'bg':'hideBg'}}">
    <view class="radioBg">
      <radio-group bindchange="radioChange">
          <view class="radioItem">
              <view class="radioName">请选择获知渠道</view>
              <view class="radioVal"><radio value="请选择获知渠道" checked/></view>
          </view>
          <view class="radioItem">
              <view class="radioName">大厅落地液晶屏广告</view>
              <view class="radioVal"><radio value="大厅落地液晶屏广告"/></view>
          </view>
          <view class="radioItem">
              <view class="radioName">其他</view>
              <view class="radioVal"><radio value="其他"/></view>
          </view>
          <view class="radioItem">
              <view class="radioName">报纸</view>
              <view class="radioVal"><radio value="报纸"/></view>
          </view>
          <view class="radioItem">
              <view class="radioName">商场超市电视广告</view>
              <view class="radioVal"><radio value="商场超市电视广告"/></view>
          </view>
          <view class="radioItem">
              <view class="radioName">公交-电视</view>
              <view class="radioVal"><radio value="公交-电视"/></view>
          </view>
          <view class="radioItem">
              <view class="radioName">餐厅 LED 屏</view>
              <view class="radioVal"><radio value="餐厅 LED 屏"/></view>
          </view>
          <view class="radioItem">
              <view class="radioName">网站</view>
              <view class="radioVal"><radio value="网站"/></view>
          </view>
          <view class="radioItem">
              <view class="radioName">电梯液晶看板</view>
              <view class="radioVal"><radio value="电梯液晶看板"/></view>
          </view>
```

```
            <view class="radioItem">
                <view class="radioName">地铁-电视</view>
                <view class="radioVal"><radio value="地铁-电视"/></view>
            </view>
        </radio-group>
    </view>
</view>
```

（2）在 form.wxss 样式文件里，给婚博会免费索票界面布局和获知渠道界面布局添加样式，具体代码如下所示。

```
.content{
  margin-top:30px;
}
.ticket{
      display: flex;
      flex-direction: row;
      width: 100%;
      text-align: center;
      margin-bottom: 30px;
}
.first{
      border:1px solid red;
      width: 45%;
      margin: 0 auto;
      background-color: #D60210;
      height: 35px;
      line-height: 35px;
      font-size: 14px;
      color: #ffffff;
}
.second{
      border:1px solid red;
      width: 45%;
      margin: 0 auto;
      background-color: #FDA6AE;
      height: 35px;
      line-height: 35px;
      font-size: 14px;
}
.item{
      margin: 10px;
```

```
    display: flex;
    flex-direction: row;
}
.name{
    width: 100px;
    text-align: right;
    height: 28px;
    line-height: 28px;
    font-size: 15px;
}
.val{
    width: 230px;
    height: 28px;
    border:1px solid #cccccc;
    border-radius: 5px;
    font-size:13px;
}
.holder{
    font-size: 13px;
    margin-left: 5px;
    color: #999999;
}
.box{
    font-size: 13px;
    width: 300px;
    margin-left: 10px;
}
.box checkbox{
    margin-right: 10px;
    margin-top: 10px;
}
.btn{
    width:220px;
    height: 45px;
    line-height: 45px;
    background-color: #D50310;
    color: #ffffff;
    border-radius: 50px;
}
.way{
```

```
    width: 230px;
    height: 30px;
    font-size: 13px;
    text-align: left;
}
.bg{
     display: block;
    background-color: #cccccc;
    width: 100%;
    height: 750px;
    position: absolute;
    top:0px;
    left:0px;
    z-index: 999;
    opacity: 0.9;
}
.radioBg{
    background-color: #ffffff;
    width: 80%;
    height: 500px;
    margin: 0 auto;
    margin-top:20px;
}
.radioItem{
    display: flex;
    flex-direction: row;
    height: 50px;
    align-items: center;
    border-bottom: 1px solid #cccccc;
}
.radioName{
    width: 90%;
}
.hideBg{
    display: none;
}
```

（3）在 form.js 业务逻辑处理文件里，初始化获知渠道并动态选择获知渠道，将表单提交的数据保存到本地，具体代码如下所示。

```
Page({
  data: {
```

```
      flag: '1',
      way: '请选择获知渠道'
  },
  selectWay: function () {
      this.setData({ flag: '0' });
  },
  radioChange: function (e) {
      console.log(e);
      var way = e.detail.value;
      this.setData({ flag: '1' });
      this.setData({ way: way });
  },
  formSubmit: function (e) {
      console.log(e);
      var ticket = e.detail.value;
      ticket.way = this.data.way;
      wx.setStorageSync('ticket', ticket);
  }
})
```

2.19 navigator 页面链接导航组件

navigator 页面链接导航组件是应用在 WXML 页面里的导航方式，navigator 页面链接导航组件的属性如表 2.18 所示。

表 2.18 navigator页面链接导航组件的属性

属　性	类　型	默认值	说　明
url	String		应用内的跳转链接
delta	Number		当open-type为‘navigateBack’时有效，表示回退的层数
open-type	String	navigate	跳转方式
hover-class	String	navigator-hover	指定点击时的样式类，当hover-class="none"时，没有点击态效果
hover-start-time	Number	50	按住后多久出现点击态，单位毫秒
hover-stay-time	Number	600	手指松开后点击态保留时间，单位毫秒

navigator 页面链接导航组件是根据 open-type 属性来设置跳转方式的，有五种跳转方式。

（1）open-type=navigate，对应 wx.navigateTo 的功能，保留当前页跳转，跳转后可以返回当前页。

示例代码如下所示。

```
<view>
    <navigator url="/pages/index/index?title=navigator" open-type="navigator">保留当前页面，跳转到应用内的某个页面</navigator>
</view>

Page({
  bindEvent:function(){
    wx.navigateTo({
      url: '/pages/index/index?title=navigator'
    })
  }
})
```

（2）open-type=redirect，对应 wx.redirectTo 的功能，若关闭当前页，则跳转时无法返回当前页。

示例代码如下所示。

```
<view>
    <navigator url="/pages/index/index?title=redirect" open-type="redirect">关闭当前页面，跳转到应用内的某个页面</navigator>
</view>

Page({
  bindEvent: function () {
    wx.redirectTo({
      url: '/pages/index/index?title=navigator'
    })
  }
})
```

（3）open-type= switchTab，对应 wx.switchTab 的功能，跳转到底部标签导航指定的页面。

示例代码如下所示。

```
<view>
    <navigator url="/pages/index/index?title=switchTab" open-type="switchTab">跳转到 tabBar 页面，并关闭其他所有非 tabBar 页面</navigator>
</view>
```

```
Page({
  bindEvent: function () {
    wx.switchTab({
      url: '/pages/index/index'
    })
  }
})
```

（4）open-type=reLaunch，对应 wx.reLaunch 的功能，关闭所有页面，打开应用内的某个页面。

示例代码如下所示。

```
<view>
   <navigator  url="/pages/index/index?title=reLaunch"  open-type="reLaunch">关闭所有页面，打开应用内的某个页面</navigator>
</view>

Page({
  bindEvent: function () {
    wx.reLaunch({
      url: '/pages/index/index'
    })
  }
})
```

（5）open-type=navigateBack，对应 wx.navigateBack 的功能，关闭当前页面，返回上一页面或多级页面。

示例代码如下所示。

```
<view>
   <navigator url="/pages/index/index" open-type="navigateBack">关闭当前页面，返回上一页面或多级页面</navigator>
</view>

Page({
  bindEvent: function () {
    wx.navigateBack({
       delta: 2
    })
  }
})
```

2.20　audio 音频组件

audio 音频组件是用来播放音乐的组件，每个 audio 音频组件都有唯一的标识 id，传入要播放音频的资源地址、音频名字、作者名字、高频封面的图片资源地址后就可以在 WXML 页面文件进行音频的播放了，audio 音频组件组件的属性如表 2.19 所示。

表 2.19　audio音频组件的属性

属　　性	类　　型	默认值	说　　明
id	String		video组件的唯一标识符
src	String		要播放音频的资源地址
loop	Boolean	false	是否循环播放
controls	Boolean	true	是否显示默认控件
poster	String		默认控件上音频封面的图片资源地址，如果controls属性值为false，则设置poster无效
name	String	未知音频	默认控件上的音频名字，如果controls属性值为false，则设置name无效
author	String	未知作者	默认控件上的作者名字，如果controls属性值为false，则设置author无效
binderror	EventHandle		当发生错误时触发error事件，detail = {errMsg: MediaError.code}
bindplay	EventHandle		当开始/继续播放时触发play事件
bindpause	EventHandle		当暂停播放时触发pause事件
bindtimeupdate	EventHandle		当播放进度改变时触发timeupdate事件，detail = {currentTime, duration}
bindended	EventHandle		当播放到末尾时触发ended事件

下面我们来看一下，如何在微信小程序的页面里进行音频的播放，如图 2.33 所示。



图 2.33　音频播放

（1）在 audio.wxml 页面文件里，使用 view、audio 组件进行界面的布局，具体代码如下所示。

```
<view style="text-align:center;margin-top:10px;margin-bottom:10px;">
  <audio poster="{{poster}}" name="{{name}}" author="{{author}}" src="{{src}}" id="myAudio" c
ontrols loop ></audio>
</view>

<view style="display:flex;flex-direction:row;">
<button type="primary" bindtap="audioPlay" style="width:70px;height:30px;line-height:30px;back
ground:#cccccc;">播放</button>
<button type="primary" bindtap="audioPause" style="width:70px;height:30px;line-height:30px;ba
ckground:#cccccc;">暂停</button>
<button type="primary" bindtap="audio14" style="width:70px;height:30px;line-height:30px;backg
round:#cccccc;">14 秒</button>
<button type="primary" bindtap="audioStart" style="width:70px;height:30px;line-height:30px;bac
kground:#cccccc;">开头</button>
</view>
```

（2）在 audio.js 业务逻辑处理文件里，初始化音频所需要的数据以及相应的事件，具体代码如下所示。

```
Page({
  onReady: function (e) {
    // 使用 wx.createAudioContext 获取 audio 上下文 context
    this.audioCtx = wx.createAudioContext('myAudio')
  },
  data: {
    poster: 'http://y.gtimg.cn/music/photo_new/T002R300x300M000003rsKF44GyaSk.jpg?max_
age=2592000',
    name: '此时此刻',
    author: '许巍',
    src: 'http://ws.stream.qqmusic.qq.com/M500001VfvsJ21xFqb.mp3?guid=ffffffff82def4af4b12b
3cd9337d5e7&uin=346897220&vkey=6292F51E1E384E06DCBDC9AB7C49FD713D632D313AC
4858BACB8DDD29067D3C601481D36E62053BF8DFEAF74C0A5CCFADD6471160CAF3E6A&f
romtag=46',
  },
  audioPlay: function () {
    this.audioCtx.play()
  },
  audioPause: function () {
    this.audioCtx.pause()
```

```
  },
  audio14: function () {
    this.audioCtx.seek(14)
  },
  audioStart: function () {
    this.audioCtx.seek(0)
  }
})
```

audio 音频组件的页面样式以及摆放的内容比较固定，如果想自定义音乐播放界面，比如类似酷狗音乐播放的界面（如图 2.34 所示），就不能使用 audio 音频组件来播放音乐了，而是需要使用 wx.playBackgroundAudio（OBJECT）这个 API。



图 2.34　酷狗音乐

它是使用后台播放器播放音乐，对于微信客户端来说，同一时间只能有一个后台音乐在播放。当用户离开小程序后，音乐将暂停播放；当用户点击“显示在聊天顶部”时，音乐不会暂停播放；若用户在其他小程序占用了音乐播放器，则原有小程序内的音乐将停止播放。

（1）在 audio-back.wxml 页面文件里，进行音乐播放界面的布局，具体代码如下所示。

```
<view class="content">
      <view class="playMusic">
       <view class="left" bindtap="detail" id="{{song.id}}">
           <image src="{{song.img}}" style="width:80px;height:80px;"></image>
       </view>
       <view class="right">
            <view class="info">
                 <view class="songInfo">
                       <view class="name">{{song.name}}</view>
                       <view>{{song.singer}}</view>
                 </view>
                 <view class="btn">
                 <block wx:if="{{playing === false}}">
                       <image src="/pages/images/play/stop.jpg" style="width:20px;height:23px;"
bindtap="play" id="{{song.id}}"></image>
                 </block>
                 <block wx:if="{{playing === true}}">
                       <image src="/pages/images/play/play.jpg" style="width:20px;height:23px;"
bindtap="stop" id="{{song.id}}"></image>
                 </block>
                       <image src="/pages/images/play/kuaijin.jpg" style="width:17px;height:18px;
"></image>
                       <image src="/pages/images/play/duilie.jpg" style="width:19px;height:19px;
"></image>
                 </view>
            </view>

       </view>
    </view>
</view>
```

（2）在 audio-back.wxss 页面样式文件里，给音乐播放界面添加样式，具体代码如下所示。

```
.playMusic{
   display: flex;
   flex-direction: row;
   border-top: 1px solid #cccccc;
   position: fixed;
   bottom: 0px;
   width: 100%;
```

```
}
.left{
   padding:10px;
}

.right{
   width:100%;
   margin-left: 10px;
   align-items: center;
}
.info{
   margin-top:30px;
   display: flex;
   flex-direction: row;
}
.songInfo{
   width: 50%;
   font-size: 13px;
}
.name{
   font-weight: bold;
}
.btn{
   width:60%;
}
.btn image{
   margin-right:15px;
}
```

（3）在 audio-back.js 业务逻辑处理文件里，初始化音乐播放数据相应的事件函数，具体代码如下所示。

```
Page({
   data: {
      playing: false,
      song: {}
   },
   onLoad: function () {
      wx.setStorageSync('songs', this.loadSongs());
      var songs = wx.getStorageSync('songs');
      this.setData({ song: songs[0] });
   },
```

```
play: function (e) {
    console.log(e);
    var id = e.currentTarget.id;
    var songs = wx.getStorageSync('songs');
    var song = new Object();
    for (var i = 0; i < songs.length; i++) {
        if (id = songs[i].id) {
            song = songs[i];
            break;
        }
    }
    console.log(song);
    var page = this;
    wx.playBackgroundAudio({
        dataUrl: song.url,
        name: song.name,
        singer: song.singer,
        coverImgUrl: song.img,
        complete: function (res) {
            page.setData({
                playing: true
            })
        }
    })
},
stop: function (e) {
    var that = this
    wx.pauseBackgroundAudio({
        success: function () {
            that.setData({
                playing: false
            })
        }
    })
},
loadSongs: function () {
    var songs = new Array();
    var song = new Object();
    song.id = 1;
    song.name = "微微一笑很倾城";
```

```
    song.singer = "杨洋";
    song.img = "http://y.gtimg.cn/music/photo_new/T002R90x90M000003RxTdZ0sJLwo.jpg"
    song.url = "http://stream.qqmusic.tc.qq.com/137903929.mp3";
    song.type = "2";
    songs.push(song);

    var song2 = new Object();
    song2.id = 2;
    song2.name = "你还要我怎样";
    song2.singer = "薛之谦";
    song2.img = "http://y.gtimg.cn/music/photo_new/T002R90x90M000000aWdOx24i3dG.jpg"
    song2.url = "http://stream.qqmusic.tc.qq.com/138549169.mp3";
    song2.type = "1";
    songs.push(song2);

    var song3 = new Object();
    song3.id = 3;
    song3.name = "演员";
    song3.singer = "薛之谦";
    song3.img = "http://y.gtimg.cn/music/photo_new/T002R90x90M000003y8dsH2wBHlo.jpg"
    song3.url = "http://stream.qqmusic.tc.qq.com/132636799.mp3";
    song3.type = "1";
    songs.push(song3);

    var song4 = new Object();
    song4.id = 4;
    song4.name = "此时此刻";
    song4.singer = "许巍";
    song4.img =
"http://y.gtimg.cn/music/photo_new/T002R300x300M000003rsKF44GyaSk.jpg?max_age=2592000"
    song4.url = "http://stream.qqmusic.tc.qq.com/132636799.mp3";
    song4.type = "1";
    songs.push(song4);

    var song5 = new Object();
    song5.id = 5;
    song5.name = "告白气球";
    song5.singer = "周杰伦";
    song5.img = "http://y.gtimg.cn/music/photo_new/T002R90x90M000003RMaRI1iFoYd.jpg";
```

```
        song5.url = "http://stream.qqmusic.tc.qq.com/137192078.mp3";
        song5.type = "1";
        songs.push(song5);

        return songs;

    }
})
```

2.21 image 图片组件

image 图片组件是用来展示图片的组件，有两种展现方式：一类是缩放模式，在缩放模式里有 4 种方式；另一类是裁剪模式，在裁剪模式里有 9 种方式，具体属性如表 2.20 所示。

表 2.20 image图片组件的属性

属　性	类　型	默认值	说　明
src	String		图片资源地址
mode	String	'scaleToFill'	图片裁剪、缩放的模式
binderror	HandleEvent		当错误发生时，发布到 AppService 的事件名，事件对象event.detail = {errMsg: 'something wrong'}
bindload	HandleEvent		当图片载入完毕时，发布到 AppService 的事件名，事件对象event.detail = {height:'图片高度px', width:'图片宽度px'}

通过 mode 属性来设置 4 种缩放模式和 9 种裁剪模式，如表 2.21 所示。

表 2.21 13 种模式

	模　式	说　明
缩放	scaleToFill	不保持纵横比缩放图片，使图片的宽高完全拉伸至填满image元素
缩放	aspectFit	保持纵横比缩放图片，使图片的长边能完全显示出来。也就是说，可以完整地将图片显示出来
缩放	aspectFill	保持纵横比缩放图片，只保证图片的短边能完全显示出来。也就是说，图片通常只在水平或垂直方向是完整的，另一个方向将被截取
缩放	widthFix	宽度不变，高度自动变化，保持原图宽高比不变
裁剪	top	不缩放图片，只显示图片的顶部区域

续表

	模　　式	说　　明
裁剪	bottom	不缩放图片，只显示图片的底部区域
裁剪	center	不缩放图片，只显示图片的中间区域
裁剪	left	不缩放图片，只显示图片的左边区域
裁剪	right	不缩放图片，只显示图片的右边区域
裁剪	top left	不缩放图片，只显示图片的左上边区域
裁剪	top right	不缩放图片，只显示图片的右上边区域
裁剪	bottom left	不缩放图片，只显示图片的左下边区域
裁剪	bottom right	不缩放图片，只显示图片的右下边区域

示例代码如下所示。

```
<view class="page">
  <view class="page__hd">
    <text class="page__title">image</text>
    <text class="page__desc">图片</text>
  </view>
  <view class="page__bd">
    <view class="section section_gap" wx:for="{{array}}" wx:for-item="item">
      <view class="section__title">{{item.text}}</view>
      <view class="section__ctn">
        <image style="width: 200px; height: 200px; background-color: #eeeeee;" mode="{{ite
m.mode}}" src="{{src}}"></image>
      </view>
    </view>
  </view>
</view>

Page({
  data: {
    array: [{
      mode: 'scaleToFill',
      text: 'scaleToFill：不保持纵横比缩放图片，使图片完全适应'
    }, {
      mode: 'aspectFit',
      text: 'aspectFit：保持纵横比缩放图片，使图片的长边能完全显示出来'
    }, {
      mode: 'aspectFill',
      text: 'aspectFill：保持纵横比缩放图片，只保证图片的短边能完全显示出来'
```

```
    }, {
      mode: 'top',
      text: 'top：不缩放图片，只显示图片的顶部区域'
    }, {
      mode: 'bottom',
      text: 'bottom：不缩放图片，只显示图片的底部区域'
    }, {
      mode: 'center',
      text: 'center：不缩放图片，只显示图片的中间区域'
    }, {
      mode: 'left',
      text: 'left：不缩放图片，只显示图片的左边区域'
    }, {
      mode: 'right',
      text: 'right：不缩放图片，只显示图片的右边区域'
    }, {
      mode: 'top left',
      text: 'top left：不缩放图片，只显示图片的左上方区域'
    }, {
      mode: 'top right',
      text: 'top right：不缩放图片，只显示图片的右上方区域'
    }, {
      mode: 'bottom left',
      text: 'bottom left：不缩放图片，只显示图片的左下方区域'
    }, {
      mode: 'bottom right',
      text: 'bottom right：不缩放图片，只显示图片的右下方区域'
    }],
    src: '../../resources/cat.jpg'
  },
  imageError: function(e) {
    console.log('image3 发生 error 事件，携带值为', e.detail.errMsg)
  }
})
```

2.22 video 视频组件

video 视频组件是用来播放视频的组件，在微信小程序页面中可以嵌入视频播放器来播

放视频。视频播放需要视频地址，同时可以控制播放控件，包括播放、暂停按钮、播放进度、时间、弹幕等控件的时间，video 视频组件默认宽度 300px、高度 225px，设置宽高需要通过 wxss 设置 width 和 height，具体属性如表 2.22 所示。

表 2.22　video视频组件的属性

属　　性	类　　型	默认值	说　　明
src	String		要播放视频的资源地址
controls	Boolean	true	是否显示默认播放控件（播放/暂停按钮、播放进度、时间）
danmu-list	Object	Array	弹幕列表
danmu-btn	Boolean	false	是否显示弹幕按钮，只在初始化时有效，不能动态变更
enable-danmu	Boolean	false	是否展示弹幕，只在初始化时有效，不能动态变更
autoplay	Boolean	false	是否自动播放
bindplay	EventHandle		当开始/继续播放时触发play事件
bindpause	EventHandle		当暂停播放时触发 pause 事件
bindended	EventHandle		当播放到末尾时触发 ended 事件
bindtimeupdate	EventHandle		播放进度变化时触发，event.detail = {currentTime: '当前播放时间'} 。触发频率应该在 250ms 一次
objectFit	String	contain	当视频大小与 video 容器大小不一致时，视频的表现形式。contain：包含，fill：填充，cover：覆盖

现在有很多的视频播放器，比如爱奇艺、腾讯视频等播放器，微信小程序也提供了视频播放功能，如图 2.35 所示。



图 2.35　视频播放

（1）在 video.wxml 页面文件里，使用 video 视频组件来进行界面布局，具体代码如下所示。

```
<view class="section tc">
  <video id="myVideo" src="http://wxsnsdy.tc.qq.com/105/20210/snsdyvideodownload?filekey=30280201010421301f0201690402534804102ca905ce620b1241b726bc41dcff44e00204012882540400&bizid=1023&hy=SH&fileparam=302c020101042530230204136ffd93020457e3c4ff02024ef202031e8d7f02030f42400204045a320a0201000400" danmu-list="{{danmuList}}" enable-danmu danmu-btn controls style="width:100%"></video>
</view>
```

（2）在 video.wxml 页面文件里，使用 video 视频组件来进行界面布局，具体代码如下所示。

```
Page({
  onReady: function (res) {
    this.videoContext = wx.createVideoContext('myVideo')
  },
  data: {
    src: '',
    danmuList: [
      {
        text: '第 1s 出现的弹幕',
        color: '#ff0000',
        time: 1
      },
      {
        text: '第 3s 出现的弹幕',
        color: '#ff00ff',
        time: 3
      }]
  }
})
```

2.23 map 地图组件

map 地图组件应用的范围已经很广了，比如滴滴打车、摩拜单车都会用到地图组件，在地图上可以看到我们要打的车或者要使用的单车，以方便我们找车。map 地图组件的具体属性如表 2.23 所示。

表 2.23　map地图组件的属性

属　　性	类　　型	默认值	说　　明
longitude	Number		中心经度
latitude	Number		中心纬度
scale	Number	16	缩放级别，取值范围为5~18
markers	Array		标记点
covers	Array		即将移除，请使用markers
autoplay	Boolean		是否自动播放
polyline	Array		路线
circles	Array		圆
controls	Array		控件
include-points	Array		缩放视野以包含所有给定的坐标点
show-location	Boolean		显示带有方向的当前定位点
bindmarkertap	EventHandle		单击标记点时触发
bindcontroltap	EventHandle		单击控件时触发
bindregionchange	EventHandle		视野发生变化时触发
bindtap	EventHandle		单击地图时触发

markers 标记点用于在地图上显示标记的位置，如表 2.24 所示。

表 2.24　marker地图标记属性

属　　性	说　　明	类型	必填	备　　注
id	标记点id	Number	否	marker点击事件回调会返回此id
latitude	纬度	Number	是	浮点数，范围 -90 ~ 90
longitude	经度	Number	是	浮点数，范围 -180 ~ 180
title	标注点名	String	否	
iconPath	显示的图标	String	是	项目目录下的图片路径，支持相对路径写法，以'/'开头则表示相对小程序根目录
rotate	旋转角度	Number	否	顺时针旋转的角度，范围 0 ~ 360，默认为 0
alpha	标注的透明度	Number	否	默认1，无透明
width	标注图标宽度	Number	否	默认为图片实际宽度
height	标注图标高度	Number	否	默认为图片实际高度

polyline 指定一系列坐标点，从数组第一项连线至最后一项，如表 2.25 所示。

表 2.25　polyline坐标点属性

属　　性	说　　明	类　型	必填	备　　注
points	经纬度数组	Array	是	[{latitude: 0, longitude: 0}]
color	线的颜色	String	否	8 位十六进制表示，后两位表示 alpha 值，如 #000000AA
width	线的宽度	Number	否	
dottedLine	是否虚线	Boolean	否	默认false

circles 在地图上会显示为圆，其属性如表 2.26 所示。

表 2.26　circles显示圆的属性

属　　性	说　　明	类　型	必　填	备　　注
latitude	纬度	Number	是	浮点数，范围 -90 ~ 90
longitude	经度	Number	是	浮点数，范围 -180 ~ 180
color	描边的颜色	String	否	8 位十六进制表示，后两位表示 alpha 值，如 #000000AA
fillColor	填充颜色	String	否	8 位十六进制表示，后两位表示 alpha 值，如 #000000AA
radius	半径	Number	是	
strokeWidth	描边的宽度	Number	否	

controls 在地图上显示控件，控件不随着地图移动，其属性如表 2.27 所示。

表 2.27　controls显示控件的属性

属　　性	说　　明	类　型	必　填	备　　注
id	控件id	Number	否	在控件单击事件回调时会返回此id
position	控件在地图的位置	Object	是	控件相对地图位置
iconPath	显示的图标	String	是	项目目录下的图片路径，支持相对路径写法，以'/'开头，则表示相对小程序根目录
clickable	是否可点击	Boolean	否	默认不可单击

position 控件位置是相对地图的位置，其属性如表 2.28 所示。

表 2.28　position控件位置的属性

属　　性	说　　明	类　　型	必填	备　　注
left	距离地图的左边界多远	Number	否	默认为0
top	距离地图的上边界多远	Number	否	默认为0

续表

属　　性	说　　明	类　　型	必填	备　　注
width	控件宽度	Number	否	默认为图片宽度
height	控件高度	Number	否	默认为图片高度

下面使用 map 地图组件用来进行摩拜单车微信小程序设计，如图 2.36 所示。



图 2.36　摩拜单车

（1）在 map.wxml 页面文件里，使用 map 地图组件进行摩拜单车界面布局设计，具体代码如下所示。

```
<view>
  <map id="map" style="width: 100%; height: 600px;" longitude="{{longitude}}" latitude="{{latitude}}" scale="14" markers="{{markers}}" controls="{{controls}}" bindcontroltap="controltap"/>
</view>
```

（2）在 map.js 业务逻辑处理文件里，在地图上放置单车、红包、我的位置以及扫码开

锁覆盖物，具体代码如下所示。

```
//index.js
//获取应用实例
var app = getApp()
var latitude, longitude
//经纬度参数
Page({
  data: {
    latitude: 22.543099,
    longitude: 114.057868,
    lockhidden: true,
    markers: [{
      //摩拜单车的位置
      iconPath: "/pages/images/mobai/bike@red.png",
      id: 1,
      latitude: 22.540069,
      longitude: 114.059878,
      width: 40,
      height: 40
    }, {
        iconPath: "/pages/images/mobai/bike@white.png",
      id: 2,
      latitude: 22.544069,
      longitude: 114.050878,
      width: 40,
      height: 40
    }, {
        iconPath: "/pages/images/mobai/bike@red.png",
      id: 3,
      latitude: 22.546069,
      longitude: 114.056878,
      width: 40,
      height: 40
    }, {
        iconPath: "/pages/images/mobai/tome.png",
      id: 4,
      latitude: 22.546099,
      longitude: 114.057868,
      width: 70,
      height: 30
```

```
    }, {
        iconPath: "/pages/images/mobai/hongbao.png",
      id: 5,
      latitude: 22.543069,
      longitude: 114.056878,
      width: 40,
      height: 40
    }, {
      // 我的位置
      iconPath: "/pages/images/mobai/lable@icon.png",
      id: 0,
      latitude: 22.543099,
      longitude: 114.057868,
      width: 40,
      height: 40
    }],
    //控件
    controls: [{
      // 我的位置控件
      id: 0,
      iconPath: "/pages/images/mobai/imgs_main_location@2x.png",
      position: {
        left: 10,
        top: 590,
        width: 50,
        height: 50
      },
      clickable: true
    }, {
      // 红包控件
      id: 1,
      iconPath: "/pages/images/mobai/hongbao.png",
      position: {
        left: 350,
        top: 525,
        width: 50,
        height: 50
      },
      clickable: true
    }, {
```

```
            //二维码控件
            id: 3,
            iconPath: "/pages/images/mobai/lock.png",
            position: {
              left: 130,
              top: 480,
              width: 120,
              height: 60
            },
            clickable: true
          }]
        },
        onLoad: function () {
          var that = this
          // 获取当前经纬度
          wx.getLocation({
            type: 'wgs84',
            success: function (res) {
              latitude = res.latitude
              longitude = res.longitude
              console.log(latitude)
              console.log(longitude)
            }
          })
        }
})
```

2.24 canvas 画布组件

canvas 画布组件可用来绘制各种形状的图形，可以把它理解成画布，在上面画什么都可以。微信小程序提供了 canvas 画布组件以及针对绘画的很多相关 API，有了这些东西，就可以随心所欲地绘制。canvas 画布组件默认宽度 300px、高度 225px，在使用的时候需要有唯一的标识。它有多个事件，包括手指触摸动作开始、手指触摸后移动、手指触摸动作结束、手指触摸动作被打断等，具体属性如表 2.29 所示。

表 2.29　canvas画布组件的属性

属　性	类　型	默认值	说　明
canvas-id	String		canvas 组件的唯一标识符
disable-scroll	Boolean	false	当在 canvas 中移动时，禁止屏幕滚动以及下拉刷新
bindtouchstart	EventHandle		手指触摸动作开始
bindtouchmove	EventHandle		手指触摸后移动
bindtouchend	EventHandle		手指触摸动作结束
bindtouchcancel	EventHandle		手指触摸动作被打断，如来电提醒、弹窗
bindlongtap	EventHandle		手指长按 500ms 之后触发，触发了长按事件后再进行移动不会触发屏幕的滚动
binderror	EventHandle		当发生错误时触发 error 事件，detail = {errMsg: 'something wrong'}

下面通过 canvas 画布组件和相关的 API 进行图形图像的绘制，如图 2.37 所示。



图 2.37　绘画

（1）在 canvas.wxml 页面文件里，使用 view 组件、canvas 组件进行页面布局设计，具

体代码如下所示。

```
<view>
  <!--画布区域-->
  <view class="canvas_area">
    <canvas canvas-id="myCanvas" class="myCanvas" disable-scroll="false" bindtouchstart="touchStart" bindtouchmove="touchMove" bindtouchend="touchEnd">
    </canvas>
  </view>
  <!--画布工具区域-->
  <view class="canvas_tools">
    <view class="box box1" bindtap="penSelect" data-param="5">钢笔</view>
    <view class="box box2" bindtap="penSelect" data-param="15">毛笔</view>
    <view class="box box3" bindtap="colorSelect" data-param="yellow">黄色</view>
    <view class="box box4" bindtap="colorSelect" data-param="blue">蓝色</view>
    <view class="box box5" bindtap="clearCanvas">擦除</view>
  </view>
</view>
```

（2）在 canvas.wxss 样式文件里，给画布以及相关按钮添加样式，具体代码如下所示。

```
.myCanvas{
  width: 100%;
  height: 500px;
}
.canvas_tools{
  position: absolute;
  bottom: 0px;
  display: flex;
  flex-direction: row;
  width: 100%;
}
.box{
  width: 60px;
  height: 60px;
  border: 1px solid #cccccc;
  border-radius: 50%;
  text-align: center;
  line-height: 60px;
  margin: 0 auto;
  background: green;
  color: #ffffff;
}
```

（3）在 canvas.js 业务逻辑处理文件里，初始化画布需要的数据以及相关事件，具体代码如下所示。

```
Page({
  data: {
    pen: 5, //画笔粗细默认值
    color: '#000000' //画笔颜色默认值
  },
  startX: 0, //保存 X 坐标轴变量
  startY: 0, //保存 Y 坐标轴变量
  isClear: false, //是否启用橡皮擦标记
  //手指触摸动作开始
  touchStart: function (e) {
    //得到触摸点的坐标
    this.startX = e.changedTouches[0].x
    this.startY = e.changedTouches[0].y
    this.context = wx.createContext()
    if (this.isClear) { //判断是否启用的橡皮擦功能：ture 表示清除，false 表示画画
      this.context.setStrokeStyle('#FFFFFF') //设置线条样式 此处设置为画布的背景颜色，橡皮擦
//原理是：擦过的地方被填充为画布的背景颜色，从而达到橡皮擦的效果
      this.context.setLineCap('round') //设置线条端点的样式
      this.context.setLineJoin('round') //设置两线相交处的样式
      this.context.setLineWidth(20) //设置线条宽度
      this.context.save();  //保存当前坐标轴的缩放、旋转、平移信息
      this.context.beginPath() //开始一个路径
      this.context.arc(this.startX, this.startY, 5, 0, 2 * Math.PI, true);  //添加一个弧形路径到当前
//路径，顺时针绘制，这里总共画了 360 度，也就是一个圆形
      this.context.fill();  //对当前路径进行填充
      this.context.restore();  //恢复之前保存过的坐标轴的缩放、旋转、平移信息
    } else {
      this.context.setStrokeStyle(this.data.color)
      this.context.setLineWidth(this.data.pen)
      this.context.setLineCap('round') //让线条圆润
      this.context.beginPath()
    }
  },

  //手指触摸后移动
  touchMove: function (e) {
    var startX1 = e.changedTouches[0].x
    var startY1 = e.changedTouches[0].y
```

```
    if (this.isClear) { //判断是否启用橡皮擦功能：ture 表示清除，false 表示画画
      this.context.save();   //保存当前坐标轴的缩放、旋转、平移信息
      this.context.moveTo(this.startX, this.startY);   //把路径移动到画布中的指定点，但不创建线条
      this.context.lineTo(startX1, startY1);   //添加一个新点，然后在画布中创建从该点到最后指定
//点的线条
      this.context.stroke();   //对当前路径进行描边
      this.context.restore()   //恢复之前保存过的坐标轴的缩放、旋转、平移信息
      this.startX = startX1;
      this.startY = startY1;
    } else {
      this.context.moveTo(this.startX, this.startY)
      this.context.lineTo(startX1, startY1)
      this.context.stroke()
      this.startX = startX1;
      this.startY = startY1;
    }

    //只是一个记录方法调用的容器，用于生成记录绘制行为的 actions 数组。与 context 不存在对
//应关系，一个 context 生成画布的绘制动作数组可以应用于多个容器
    wx.drawCanvas({
      canvasId: 'myCanvas',
      reserve: true,
      actions: this.context.getActions() //  获取绘图动作数组
    })

  },

  //手指触摸动作结束

  touchEnd: function () {
  },

  //启动橡皮擦方法
  clearCanvas: function () {
    if (this.isClear) {
      this.isClear = false;
    } else {
      this.isClear = true;
    }
  },
```

```
    penSelect: function (e) { //更改画笔大小的方法
        console.log(e.currentTarget);
        this.setData({ pen: parseInt(e.currentTarget.dataset.param) });
        this.isClear = false;
    },
    colorSelect: function (e) { //更改画笔颜色的方法
        console.log(e.currentTarget);
        this.setData({ color: e.currentTarget.dataset.param });
        this.isClear = false;
    }
})
```

第 3 章　微信小程序必备 API 技能

微信小程序提供了很多 API 接口，通过这些接口可以满足实际项目开发过程的需求。这些 API 的接口类别有：网络请求、媒体操作、文件处理、数据缓存、获取位置、设备信息、界面、第三方平台、开发接口、数据分析和拓展接口，这些 API 有一些是经常会用到的，需要掌握，另外一些只要懂得它的原理，在需要的时候查询 API 能进行使用就可以了。

3.1　wx.request 发起网络请求

微信小程序可以理解为客户端，就像 Android 和 iOS 一样都是属于客户端软件，它们都有自己的后台，有的后台语言用的是 PHP、Java 等语言，这些语言用来处理后台业务逻辑，给微信小程序提供相应的数据，这时候请求后台的数据就需要用到 wx.request 发起网络请求，从后台获取数据。wx.request(OBJECT)发起网络请求 API 的 OBJECT 参数说明如表 3.1 所示。

表 3.1　wx.request(OBJECT)参数说明

属　　性	类　　型	必填	说　　明
url	String	是	开发者服务器接口地址
data	Object、String	否	请求的参数
header	Object	否	设置请求的 header ，header 中不能设置 Referer
method	String	否	默认为GET，有效值：OPTIONS、GET、HEAD、POST、PUT、DELETE、TRACE、CONNECT
dataType	String	否	默认为JSON。如果设置了dataType为JSON，则会尝试对响应的数据做一次JSON.parse
success	Function	否	收到开发者服务成功返回的回调函数，res = {data: '开发者服务器返回的内容'}
fail	Function	否	接口调用失败的回调函数
complete	Function	否	接口调用结束的回调函数（无论调用成功、失败都会执行）

success 返回参数说明。

（1）data：开发者服务器返回的数据。

（2）statusCode：开发者服务器返回的状态码。

（3）header：开发者服务器返回的 HTTP Response Header。

wx.request 发起网络请求的 URL 服务器地址，必须在微信公众平台的设置模块里有开发设置，可以配置服务器域名，如图 3.1 所示。

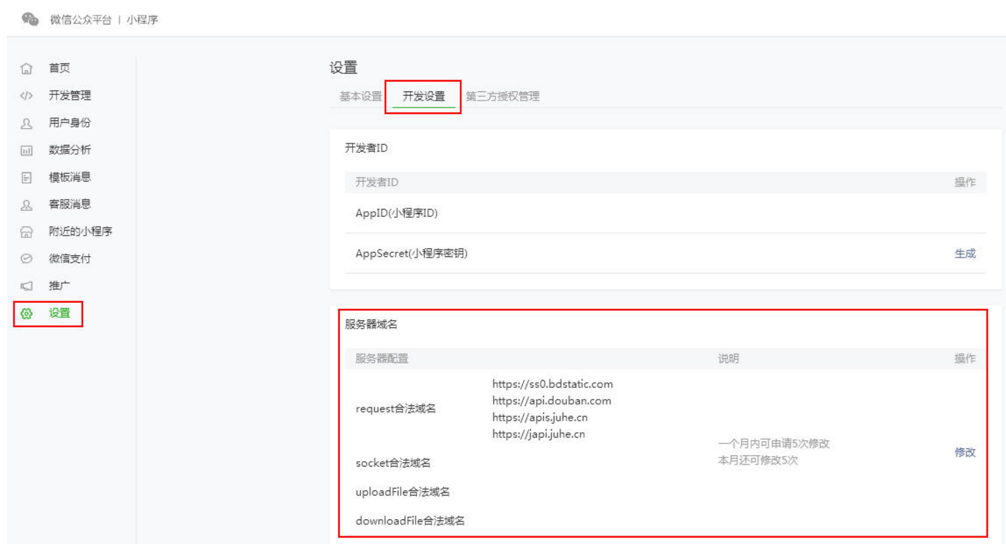


图 3.1　服务器域名配置

从图 3.1 中可以看出，request 发起网络请求的域名、socket 会话连接请求的域名、uploadFile 上传文件请求的域名、downloadFile 下载文件请求的域名都是需要配置的，如果没有配置就直接使用则是非法的，即获取不到后台数据。下面可以演示一下合法路径和非法路径的使用效果。

1．请求没有配置服务器域名

先申请一个 AppId 小程序 ID，现在微信公众平台允许以个人名义申请小程序 ID，申请完小程序 ID 后，创建一个带有 AppId 的项目，这样才能自动关联到服务器域名配置的合法路径，如果服务器域名里没有找到请求的 URL 域名，则微信小程序会认为这是非法请求。

从中央气象台获取北京天气情况，示例代码如下所示。

```
Page({
  onLoad: function () {
    wx.request({
```

```
            url: 'http://www.weather.com.cn/data/sk/101010100.html',
            data: {
                type: ''
            },
            method: 'GET',
            success: function (res) {
                console.log(res);
            }
        })
    }
})
```

请求的服务器 URL 地址 http://www.weather.com.cn/data/sk/101010100.html，在微信公众平台服务器域名是没有配置的，如果域名存在，则允许访问，否则不允许访问，因为会被认为是非法的，如图 3.2 所示。

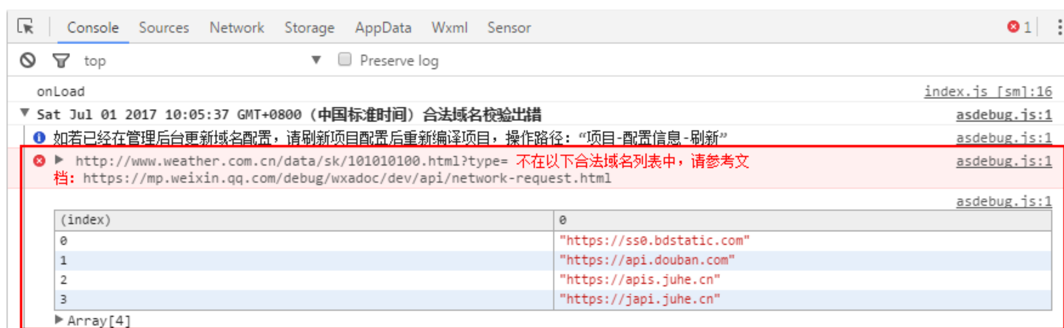


图 3.2　合法域名

2．请求配置服务器域名

在服务器域名里配置了豆瓣电影的域名 https://api.douban.com，下面获取豆瓣电影列表，具体代码如下所示。

```
Page({
    onLoad: function () {
        wx.request({
            url: 'https://api.douban.com/v2/movie/in_theaters',
            data: {
            },
            method: 'GET',
            header: {
                'content-type': 'json'
            },
```

```
            success: function (res) {
                console.log(res);
            }
        })
    }
})
```

由于请求的服务器 URL 地址是合法的，因此可以获取豆瓣电影列表，如图 3.3 所示。

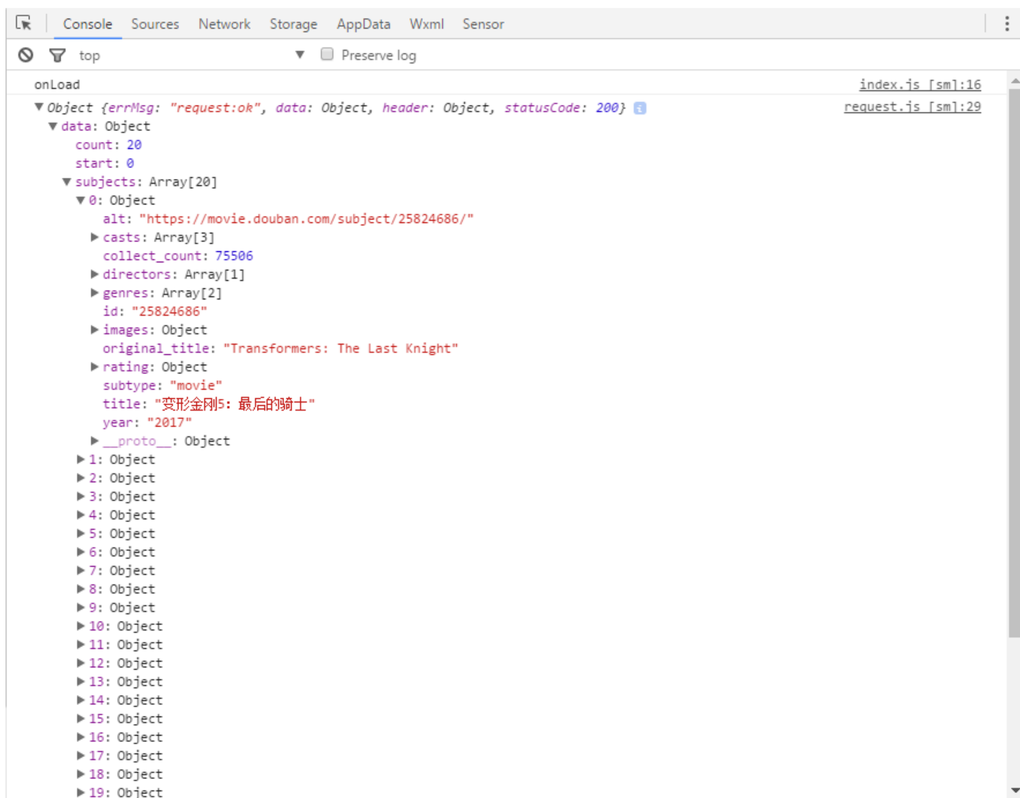


图 3.3　豆瓣电影列表

注意：content-type 默认为 'application/json'，客户端的 HTTPS TLS 版本为 1.2，但 Android 的部分机型还未支持 TLS 1.2，所以请确保 HTTPS 服务器的 TLS 版本支持 1.2 及以下版本。要注意 method 的 value 必须为大写（例如 GET）；url 中不能有端口；request 的默认超时时间和最大超时时间都是 60s，equest 的最大并发数是 5；网络请求的 referer 是不可以设置的，格式固定为 https://servicewechat.com/{appid}/{version}/page-frame.html，

其中{appid} 为小程序的 AppId，{version} 为小程序的版本号，版本号为 0 表示这是开发版。

3.2 wx.uploadFile 上传文件

wx.uploadFile 上传文件这个 API 可以将文件上传到后台服务器，比如可以将图片上传给后台服务器，然后客户端发起一个 HTTPS POST 请求，根据后台服务器地址 URL 将相应的内容进行传输，wx.uploadFile(OBJECT)的参数说明如表 3.2 所示。

表 3.2 wx.uploadFile(OBJECT)的参数说明

属　　性	类　　型	必填	说　　明
url	String	是	开发者服务器URL
filePath	String	是	要上传文件资源的路径
name	String	是	文件对应的key，开发者在服务器端通过这个key可以获取文件二进制内容
header	Object	否	HTTP 请求header，header 中不能设置referer
formData	Object	否	HTTP 请求中其他额外的 for，data
success	Function	否	接口调用成功的回调函数
fail	Function	否	接口调用失败的回调函数
complete	Function	否	接口调用结束的回调函数（无论调用成功、失败都会执行）

上传成功后，success 返回两个参数。

（1）data：开发者服务器返回的数据。

（2）statusCode：HTTP 状态码，200 代表请求已成功、404 代表请求找不到，等等。示例代码如下所示。

```
Page({
   onLoad:function(){
     wx.chooseImage({
       success: function (res) {
         var tempFilePaths = res.tempFilePaths //图片上传的临时路径，它是一个数组
         wx.uploadFile({
           url: 'http://example.weixin.qq.com/upload', //上传到后台服务器地址
           filePath: tempFilePaths[0],//上传图片的地址
           name: 'file',//上传文件的名称，会根据这个名称，找到上传的内容
           formData: {
```

```
                    'user': 'test' //传递的参数
                },
                success: function (res) {
                    var data = res.data //上传成功后返回的数据
                    console.log(data); //打印回传的数据
                }
            })
        }
    });
  }
})
```

注意：wx.uploadFile 上传文件最大并发数是 10 个，默认超时时间和最大超时时间都是 60s，可以在 app.js 里配置超时时间。wx.uoloadFile 上传文件请求的 URL 地址需要在微信公众平台进行配置，这才是合法的 URL 地址。

3.3　wx.downloadFile 下载文件

微信小程序不仅提供了 wx.uploadFile 上传文件，同时还提供了 wx.downloadFile 下载文件，即可以从后台服务器下载文件，然后客户端直接发起一个 HTTP GET 请求，返回文件的本地临时路径。wx.downloadFile(OBJECT)下载文件的参数说明如表 3.3 所示。

表 3.3　wx.downloadFile(OBJECT)的参数说明

属　　性	类　　型	必填	说　　明
url	String	是	开发者服务器 URL
filePath	String	是	要下载文件资源的路径
name	String	是	文件对应的key，开发者在服务器端通过这个key可以获取文件二进制内容
header	Object	否	HTTP请求header，header中不能设置referer
formData	Object	否	HTTP请求中其他额外的formdata
success	Function	否	接口调用成功的回调函数
fail	Function	否	接口调用失败的回调函数
complete	Function	否	接口调用结束的回调函数（无论调用成功、失败都会执行）

下载成功后，会获取到下载文件的临时路径，在小程序本次启动期间可以正常使用。如果需要持久保存，则必须主动调用 wx.saveFile，在小程序下次启动时才能访问得到。

下面从百度图片里，下载一张图片到微信小程序中并显示出来，如图 3.4 所示。



图 3.4　图片（源自百度）

（1）在微信公众平台的服务器域名上配置请求的服务器域名，https://ss2.bdstatic.com，配置过后才是合法的域名，才能下载这个域名下的图片。

（2）在 downloadFile.wxml 页面文件里，使用 image 图片组件进行布局，让下载下来的图片在微信小程序里显示出来，具体代码如下所示。

```
<image src="{{imgUrl}}" style="width:100%;"></image>
```

（3）在 downloadFile.js 业务逻辑处理文件里，使用 wx.downloadFile 下载文件 API 将图片从百度图片里下载出来，然后在小程序里显示出来，具体代码如下所示。

```
Page({
  data:{
    imgUrl:''
  },
  onLoad:function(){
    var page = this;
    wx.downloadFile({
      url:
"https://ss2.bdstatic.com/70cFvnSh_Q1YnxGkpoWK1HF6hhy/it/u=49292017,22064401&fm=28&gp=0.jpg",
      type: 'image', //  下载资源的类型，用于客户端识别处理，有效值：image/audio/video
      success: function (res) {
        console.log(res);
```

```
            var tempPath = res.tempFilePath;
            page.setData({ imgUrl: tempPath });
        }
    })

  }
})
```

界面效果如图 3.5 所示。



图 3.5　显示图片

注意：wx.downloadFile 文件下载最大并发限制是 10 个；默认超时时间和最大超时时间都是 60s，超时时间可以在 app.js 里进行配置。网络请求的 referer 是不可以设置的，格式固定为 https://servicewechat.com/{appid}/{version}/page-frame.html，其中 {appid} 为小程序的 AppId，{version} 为小程序的版本号，版本号为 0 时表示为开发版。

3.4　WebSocket 会话连接

WebSocket 解决了双向通信的问题，传统的浏览器访问都是单向通信：只能从浏览器发起请求，后台服务器接收请求后返回数据，如果有需求则从后台直接向浏览器推数据，

可能会采用轮询机制，这会带来连接的频繁建立与断开的性能损耗。显然，WebSocket 对于双向通信是比较好的解决方式，H5 也已经支持 WebSocket 双向通信，不仅 IE，其他浏览器对它的支持也很好，现在微信小程序也支持 WebSocket，对于以后双向通信来说，WebSocket 是一个不错的解决方案。

微信小程序使用 wx.connectSocket 创建一个 WebSocket 连接，一个微信小程序同时只能有一个 WebSocket 连接，如果当前已存在一个 WebSocket 连接，则会自动关闭该连接，并重新创建一个 WebSocket 连接。

由于 WebSocket 拥有双向通信的功能，因而可以使用它来创建聊天，就像微信聊天或者 QQ 聊天一样，如图 3.6 所示。

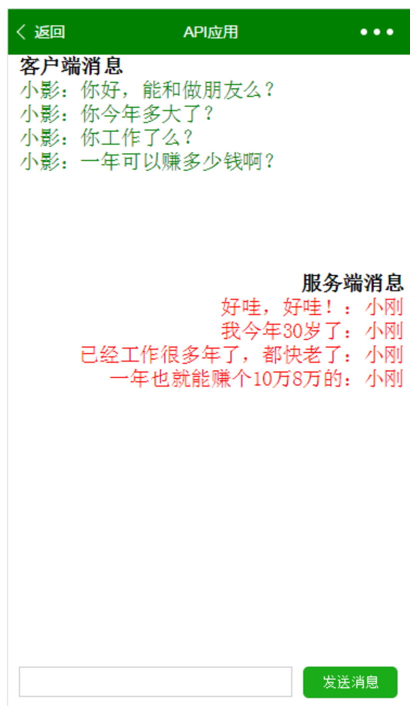


图 3.6　WebSocket 会话聊天

（1）在 WebSocket.wxml 页面文件里，进行聊天会话的布局设计，具体代码如下所示。

```
<view  class="content">
<view  style="height:200px;margin-left:10px;">
<view  style="font-weight:bold;">客户端消息</view>
<block  wx:for="{{sendMsg}}"  wx:for-item="item1">
<view  style="color:green">小影：{{item1}}</view>
```

```
</block>
</view>

<view style="height:200px;margin-right:10px;text-align:right;">
<view style="font-weight:bold;width:100%;text-align:right;">服务端消息</view>
<block wx:for="{{resData}}" wx:for-item="item2">
<view style="color:red">{{item2}}：小刚</view>
</block>
</view>

<view style="display:flex;flex-direction:row;margin:10px;position:absolute;bottom:0px;">
<input type="text" name="msg"  bindblur="getMsg" style="width:250px;border:1px solid #cccc
cc;margin-right:10px;"/>
<button type="primary" size="mini" bindtap="send">发送消息</button>
</view>
</view>
```

（2）在 WebSocket.js 业务逻辑处理文件里，编写客户端代码，包括建立连接、监听连接、关闭连接等内容，具体代码如下所示。

```
Page({
  data: {
    msg: '',
    sendMsg: [],
    socketOpen: false,
    resData: []
  },
  onLoad:function(){
    var page = this;
     wx.connectSocket({//创建一个连接
      url: 'ws://localhost:8215/wxapp/getServer',
        header: {
        'content-type': 'application/json'
      },
      method: "GET"
    });
    wx.onSocketOpen(function (res) {//监听连接已打开，可以进行通信
      console.log(res);
      page.setData({ socketOpen: true });
      console.log('WebSocket 连接已打开！')
    });
    wx.onSocketError(function (res) {//监听连接发生错误
```

```
        console.log('WebSocket 连接打开失败，请检查！')
      })
    },
    send: function (e) {
      if (this.data.socketOpen) {
        console.log(this.data.socketOpen);
        wx.sendSocketMessage({
          data: this.data.msg
        });
        var sendMsg = this.data.sendMsg;
        sendMsg.push(this.data.msg);
        this.setData({ sendMsg: sendMsg });
        var page = this;
        wx.onSocketMessage(function (res) {//监听连接可以发送消息
          var resData = page.data.resData;
          resData.push(res.data);
          page.setData({ resData: resData });
          console.log(resData);
          console.log('收到服务器内容：' + res.data)
        })
      } else {
        console.log('WebSocket 连接打开失败，请检查！');
      }
    },
    onHide:function(){
      wx.closeSocket();//关闭连接
      wx.onSocketClose(function (res) {//监听连接已关闭
        console.log('WebSocket 已关闭！')
      });
    }
})
```

（3）后台服务器也采用 WebSocket 编写服务器端代码，具体代码如下所示。

```
package com.xiaogang.app.servlet;

import java.io.IOException;

import javax.websocket.OnClose;
import javax.websocket.OnError;
import javax.websocket.OnMessage;
import javax.websocket.OnOpen;
```

```
import javax.websocket.RemoteEndpoint;
import javax.websocket.Session;
import javax.websocket.server.ServerEndpoint;



@ServerEndpoint("/getServer")
public class WebsocketServer{

    @OnOpen
    public void onOpen(Session session) {
            System.out.println("sessionId="+session.getId());
            final RemoteEndpoint.Basic basic =  session.getBasicRemote();
        try {
                basic.sendText("会话建立成功！！！");
            } catch (IOException e) {
                e.printStackTrace();
            }
        Thread t1=new Thread(new Runnable() {

                @Override
                public void run() {
                    try {
                        Thread.currentThread();
                        Thread.sleep(8000);
                        basic.sendText("server get you a msg: what your name?");
                    } catch (InterruptedException e) {
                        e.printStackTrace();
                    } catch (IOException e) {
                        e.printStackTrace();
                    }
                }
            });
        t1.start();
    }
    /**
     * 收到客户端消息时触发
     * @param relationId
     * @param userCode
     * @param message
```

```
     * @return
    */
    @OnMessage
     public String onMessage(Session session,String message) {
             System.out.println("pathParams:"+session.getPathParameters());
           System.out.println("requestParams"+session.getRequestParameterMap());
           return      "Got you msg !"+message;
    }

    /**
     * 异常时触发
     * @param relationId
     * @param userCode
     * @param session
    */
    @OnError
     public void onError(Throwable throwable,Session session) {
             System.out.println("pathParams:"+session.getPathParameters());
            System.out.println("requestParams"+session.getRequestParameterMap());
            System.out.print("onError"+throwable.toString());
    }

    /**
     * 关闭连接时触发
     * @param relationId
     * @param userCode
     * @param session
    */
    @OnClose
     public void onClose(Session session) {
             System.out.println("pathParams:"+session.getPathParameters());
                System.out.println("requestParams"+session.getRequestParameterMap());
             System.out.print("onClose ");
    }
}
```

这样就可以在客户端发送信息了，服务端可以返回消息，不进行聊天的时候，可以关闭连接。

微信小程序 WebSocket 会话调用详解。

（1）微信小程序使用 wx.connectSocket 创建一个会话连接，wx.connectSocket(OBJECT)

的参数说明如表 3.4 所示。

表 3.4　wx. connectSocket(OBJECT)的参数说明

属　　性	类　　型	必填	说　　明
url	String	是	开发者服务器URL
data	Object	否	请求的数据
header	Object	否	HTTP请求header，header中不能设置Referer
method	String	否	默认是GET，有效值为：OPTIONS、GET、HEAD、POST、PUT、DELETE、TRACE、CONNECT
success	Function	否	接口调用成功的回调函数
fail	Function	否	接口调用失败的回调函数
complete	Function	否	接口调用结束的回调函数（无论调用成功、失败都会执行）

示例代码如下所示。

```
onLoad:function(){
    var page = this;
     wx.connectSocket({//创建一个连接
       url: 'ws://localhost:8215/wxapp/getServer',
         header: {
         'content-type': 'application/json'
       },
       method: "GET"
    });
    wx.onSocketOpen(function (res) {//监听连接已打开，可以进行通信
       console.log(res);
       page.setData({ socketOpen: true });
       console.log('WebSocket 连接已打开！')
    });
    wx.onSocketError(function (res) {//监听连接发生错误
       console.log('WebSocket 连接打开失败，请检查！')
    })
  }
```

（2）微信小程序在使用完 wx.connectSocket 创建一个连接后，就可以开始监听会话连接是否打开，可以使用 wx.onSocketOpen 来监听，示例代码如下所示。

```
onLoad:function(){
    var page = this;
     wx.connectSocket({//创建一个连接
       url: 'ws://localhost:8215/wxapp/getServer',
         header: {
```

```
            'content-type': 'application/json'
        },
        method: "GET"
    });
    wx.onSocketOpen(function (res) {//监听连接已打开，可以进行通信
        console.log(res);
        page.setData({ socketOpen: true });
        console.log('WebSocket 连接已打开！')
    });
    wx.onSocketError(function (res) {//监听连接发生错误
        console.log('WebSocket 连接打开失败，请检查！')
    })
}
```

（3）微信小程序同时提供监听连接错误，如果连接没有打开，则会使用 wx.onSocketError 来监听，示例代码如下所示。

```
onLoad:function(){
    var page = this;
     wx.connectSocket({//创建一个连接
        url: 'ws://localhost:8215/wxapp/getServer',
            header: {
            'content-type': 'application/json'
        },
        method: "GET"
    });
    wx.onSocketOpen(function (res) {//监听连接已打开，可以进行通信
        console.log(res);
        page.setData({ socketOpen: true });
        console.log('WebSocket 连接已打开！')
    });
    wx.onSocketError(function (res) {//监听连接发生错误
        console.log('WebSocket 连接打开失败，请检查！')
    })
}
```

（4）微信小程序监听会话连接打开正常，这时就可以发送消息进行通信，即用 wx.sendSocketMessage(OBJECT)发送消息，使用 wx.onSocketMessage 监听服务器返回来的数据，示例代码如下所示。

```
send: function (e) {
    if (this.data.socketOpen) {
        console.log(this.data.socketOpen);
```

```
            wx.sendSocketMessage({//发送消息
               data: this.data.msg
            });
            var sendMsg = this.data.sendMsg;
            sendMsg.push(this.data.msg);
            this.setData({ sendMsg: sendMsg });
            var page = this;
            wx.onSocketMessage(function (res) {//监听服务器返回来的数据
               var resData = page.data.resData;
               resData.push(res.data);
               page.setData({ resData: resData });
               console.log(resData);
               console.log('收到服务器内容：' + res.data)
            })
         } else {
            console.log('WebSocket 连接打开失败，请检查！');
         }
      }
```

（5）微信小程序会话聊天结束后，可以使用 wx.closeSocket()关闭连接，同时可以使用 wx.onSocketClose(CALLBACK)监听连接关闭，示例代码如下所示。

```
onHide:function(){
    wx.closeSocket();//关闭连接
    wx.onSocketClose(function (res) {//监听连接已关闭
       console.log('WebSocket 已关闭！')
    });
 }
```

3.5　wx.chooseImage 选择图片

微信小程序使用 wx.chooseImage 从本地相册选择图片或使用相机拍照，选择成功后会返回文件列表的临时路径，wx.chooseImage(OBJECT)参数说明如表 3.5 所示。

表 3.5　wx. chooseImage(OBJECT)参数说明

属　　性	类　　型	必　填	说　　明
count	Number	否	最多可以选择的图片张数，默认9
sizeType	StringArray	否	original 原图，compressed 压缩图，默认二者都有

续表

属　　性	类　　型	必　填	说　　明
sourceType	StringArray	否	album 从相册选图，camera 使用相机，默认二者都有
success	Function	否	成功则返回图片的本地文件路径列表 tempFilePaths
fail	Function	否	接口调用失败的回调函数
complete	Function	否	接口调用结束的回调函数（无论调用成功、失败都会执行）

Success 返回两个参数：一个是 tempFilePaths 图片的本地文件路径列表，另一个是 tempFiles 图片的本地文件列表，每一项是一个 File 对象。返回的路径都是文件的临时路径，在小程序本次启动期间可以正常使用，如需持久保存，则要主动调用 wx.saveFile，在小程序下次启动时才能访问。

示例代码如下所示。

```
wx.chooseImage({
  count: 1, // 默认 9
  sizeType: ['original', 'compressed'], // 可以指定是原图还是压缩图，默认二者都有
  sourceType: ['album', 'camera'], // 可以指定来源是相册还是相机，默认二者都有
  success: function (res) {
    // 返回选定照片的本地文件路径列表，tempFilePath 可以作为 img 标签的 src 属性显示图片
    var tempFilePaths = res.tempFilePaths
  }
})
```

3.6　wx.previewImage 预览图片

微信小程序提供了图片预览功能，使用 wx.previewImage 可以预览多张图片，wx.previewImage(OBJECT)参数说明如表 3.6 所示。

表 3.6　wx. previewImag(OBJECT)参数说明

属　　性	类　　型	必填	说　　明
current	String	否	当前显示图片的链接，不填则默认为 urls 的第一张
urls	StringArray	是	需要预览的图片链接列表
success	Function	否	接口调用成功的回调函数
fail	Function	否	接口调用失败的回调函数
complete	Function	否	接口调用结束的回调函数（无论调用成功、失败都会执行）

wx.previewImage(OBJECT)参数里有 current 属性，如果这个参数值为空，则默认显示第一张图片；如果这个参数值不为空，则显示这张图片，示例代码如下所示。

```
Page({
  onLoad:function(){
    wx.previewImage({
      current: 'http://img02.tooopen.com/images/20150928/tooopen_sy_143912755726.jpg',
 // 当前显示图片的 http 链接
      urls: [
        "http://img02.tooopen.com/images/20150928/tooopen_sy_143912755726.jpg",
        "http://img06.tooopen.com/images/20160818/tooopen_sy_175866434296.jpg",
        "http://img06.tooopen.com/images/20160818/tooopen_sy_175833047715.jpg"
      ] // 需要预览的图片 http 链接列表
      success:(){
          //成功后回调
      }
    })
  }
})
```

3.7　wx.getImageInfo 获取图片信息

微信小程序可以获取指定图片的信息，包括图片的宽度、高度以及图片的本地路径，可以使用 wx.getImageInfo 来获取这些图片信息。wx.getImageInfo(OBJECT)图片的参数说明如表 3.7 所示。

表 3.7　wx.getImageInfo(OBJECT)参数说明

属　　性	类　　型	必填	说　　明
src	StringArray	是	图片的路径，可以是相对路径、临时文件路径、存储文件路径或网络图片路径
success	Function	否	接口调用成功的回调函数
fail	Function	否	接口调用失败的回调函数
complete	Function	否	接口调用结束的回调函数（无论调用成功还是失败都会执行）

wx.getImageInfo 调用成功后会返回三个参数：width 图片宽度，单位 px；height 图片高度，单位 px；path 返回图片的本地路径。

示例代码如下所示。

```
Page({
  onLoad:function(){
    wx.getImageInfo({
      src: 'https://ss2.bdstatic.com/70cFvnSh_Q1YnxGkpoWK1HF6hhy/it/u=49292017,2206440
1&fm=28&gp=0.jpg',
      success: function (res) {
        console.log('图片宽度：'+res.width)
        console.log('图片高度：'+res.height)
        console.log('图片路径：' + res.path)
      }
    })
  }
})
```

打印的图片信息如图 3.7 所示。

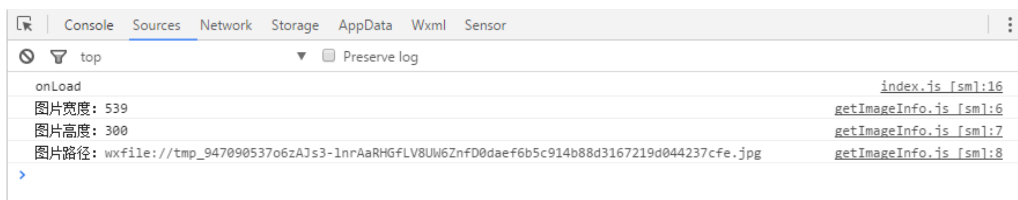


图 3.7 打印的图片信息

3.8 wx.saveImageToPhotosAlbum 保存图片到相册

微信小程序支持将图片保存到系统相册里，但前提是需要用户授权才可以保存到系统相册里，wx.saveImageToPhotosAlbum(OBJECT)参数说明如表 3.8 所示。

表 3.8 wx. saveImageToPhotosAlbum(OBJECT)参数说明

属 性	类 型	必填	说 明
filePath	StringArray	是	图片文件路径，可以是临时文件路径，也可以是永久文件路径
success	Function	否	接口调用成功的回调函数
fail	Function	否	接口调用失败的回调函数
complete	Function	否	接口调用结束的回调函数（无论调用成功、失败都会执行）

wx.saveImageToPhotosAlbum 调用成功后会返回来一个参数：errMsg 调用结果。

示例代码如下所示。

```
Page({
  data: {
    imgUrl: ''
  },
  onLoad: function () {
    var page = this;
    wx.downloadFile({
      url: "https://ss2.bdstatic.com/70cFvnSh_Q1YnxGkpoWK1HF6hhy/it/u=49292017,2206440
1&fm=28&gp=0.jpg",
      type: 'image',
      success: function (res) {
        console.log(res);
        var tempPath = res.tempFilePath;

        wx.saveImageToPhotosAlbum({
          filePath: tempPath,
          success:function(res){
            console.log(res);
          }
        })
      }
    })

  }
})
```

打印信息如图 3.8 所示。

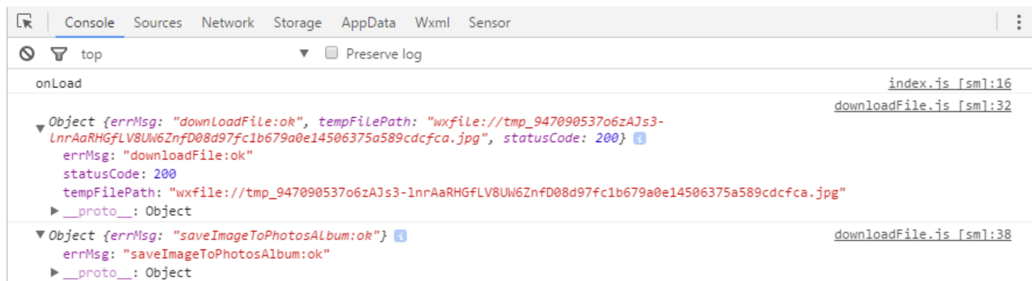


图 3.8　打印信息

3.9 wx.startRecord 开始录音和 wx.stopRecord 结束录音

微信小程序提供了两个 API 录音接口，一个是 wx.startRecord 开始录音，另一个是 wx.stopRecord 结束录音。开始录音时，微信小程序主动调用 wx.stopRecord 这个接口，或者录音超过 1 分钟时自动结束录音，返回录音文件的临时文件路径。当用户离开小程序时，此接口无法调用。wx.startRecord(OBJECT)参数说明如表 3.9 所示。

表 3.9 wx.startRecord(OBJECF)参数说明

属　　性	类　　型	必填	说　　明
success	Function	否	接口调用成功的回调函数
fail	Function	否	接口调用失败的回调函数
complete	Function	否	接口调用结束的回调函数（无论调用成功、失败都会执行）

接口调用成功后会返回 tempFilePath 录音文件的临时路径，示例代码如下所示。

（1）在 startRecord.wxml 页面文件里放置两个按钮，一个是开始录音，另外一个是结束录音，具体代码如下所示。

```
<view style="width:100%;text-align:center;margin-top:30px;">
   <button type="primary" size="mini" style="margin-right:10px;" bindtap="startRecord">开始录音</button>
   <button type="primary" size="mini" bindtap="stopRecord">结束录音</button>
</view>
<view style="margin:10px;">
filePath:{{filePath}}
</view>
```

（2）在 startRecord.js 业务逻辑处理文件里，提供开始录音和结束录音的两个函数，具体代码如下所示。

```
Page({
  data:{
    filePath:''
  },
  startRecord:function(){
    var page = this;
    wx.startRecord({//开始录音
      success: function (res) {
        var tempFilePath = res.tempFilePath
        page.setData({ filePath: tempFilePath});
```

```
        },
        fail: function (res) {
          console.log("录音失败");
        }
      })
    },
    stopRecord:function(){
      wx.stopRecord();//结束录音
      var page = this;
      console.log(page.data.filePath);
    }
})
```

3.10　voice 音频播放控制

对于音频播放控制，微信小程序提供了三个 API 接口：wx.playVoice 播放语音、wx.pauseVoice 暂停播放语音、wx.stopVoice 结束播放语音。

（1）wx.playVoice 播放语音，同时间只允许一个语音文件播放，如果前一个语音文件还没播放完，那么将中断前一个语音播放。

（2）wx.pauseVoice 暂停播放语音，再次调用 wx.playVoice 播放同一个文件时，会从暂停处开始播放。如果想从头开始播放，则需要先调用 wx.stopVoice。

wx.playVoice(OBJECT)参数说明如表 3.10 所示。

表 3.10　wx.playVoice(OBJECT)参数说明

属　　性	类　　型	必填	说　　明
filePath	String	是	需要播放的语音文件的文件路径
success	Function	否	接口调用成功的回调函数
fail	Function	否	接口调用失败的回调函数
complete	Function	否	接口调用结束的回调函数（无论调用成功、失败都会执行）

（1）在 voice.wxml 页面文件里提供了三个按钮：开始播放、暂停播放、停止播放，具体代码如下所示。

```
<view style="display:flex;flex-direction:row;margin-top:30px;">
<button type="primary" bindtap="startVoice" style="width:70px;height:30px;line-height:30px;bac
kground:#cccccc;">播放</button>
<button type="primary" bindtap="pauseVoice" style="width:70px;height:30px;line-height:30px;ba
```

```
ckground:#cccccc;">暂停</button>
<button type="primary" bindtap="stopVoice" style="width:70px;height:30px;line-height:30px;back
ground:#cccccc;">结束</button>
</view>
```

（2）在 voice.js 业务逻辑处理文件里，提供了开始播放、暂停播放、停止播放三个按钮的相关函数，具体代码如下所示。

```
Page({
  startVoice:function(){
    wx.startRecord({
      success: function (res) {
        var tempFilePath = res.tempFilePath
        wx.playVoice({//开始播放
          filePath: tempFilePath,
          complete: function () {
          }
        })
      }
    })
  },
  pauseVoice:function(){
    wx.pauseVoice();//暂停播放
  },
  stopVoice:function(){
    wx.stopVoice();//停止播放
  }
})
```

3.11 wx.createAudioContext 音频组件控制

微信小程序使用 wx.createAudioContex 音频控制组件创建了一个音频播放的 audioContext 上下文环境。audioContext 通过 audioId 与一个<audio/>组件绑定，通过它可以操作对应的 <audio/> 组件。audioContext 对象提供了四个方法：setSrc 音频的地址、play 播放、pause 暂停、seek 跳转到指定位置（单位 s）。

示例代码如下所示。

```
<view style="text-align:center;margin-top:10px;margin-bottom:10px;">
  <audio  src="{{src}}" id="myAudio" controls loop ></audio>
```

```
</view>

<view style="display:flex;flex-direction:row;">
<button type="primary" bindtap="audioPlay" style="width:70px;height:30px;line-height:30px;back
ground:#cccccc;">播放</button>
<button type="primary" bindtap="audioPause" style="width:70px;height:30px;line-height:30px;ba
ckground:#cccccc;">暂停</button>
<button type="primary" bindtap="audio14" style="width:70px;height:30px;line-height:30px;backg
round:#cccccc;">14 秒</button>
</view>

Page({
  onReady: function (e) {
    // 使用 wx.createAudioContext 获取 audio 上下文 context
    this.audioCtx = wx.createAudioContext('myAudio')
    this.audioCtx.setSrc('http://ws.stream.qqmusic.qq.com/M500001VfvsJ21xFqb.mp3?guid=ffffff
ff82def4af4b12b3cd9337d5e7&uin=346897220&vkey=6292F51E1E384E06DCBDC9AB7C49FD7
13D632D313AC4858BACB8DDD29067D3C601481D36E62053BF8DFEAF74C0A5CCFADD647
1160CAF3E6A&fromtag=46')
    this.audioCtx.play()
  },
  data: {
    src: ''
  },
  audioPlay: function () {
    this.audioCtx.play()
  },
  audioPause: function () {
    this.audioCtx.pause()
  },
  audio14: function () {
    this.audioCtx.seek(14)
  }
})
```

3.12　wx.playBackgroundAudio 播放音乐

微信小程序的后台音乐播放器使用的是 wx.playBackgroundAudio 这个 API 接口，直接

调用这个接口就可以进行音乐播放，而不用 audio 音频组件。这也是很多播放器采用的一种方式，因为 audio 音频组件样式比较固定，不能进行个性化的定制。对于微信客户端来说，只能同时有一个后台音乐在播放。当用户离开小程序后，音乐将暂停播放；当用户单击“显示在聊天顶部”时，音乐不会暂停播放；当用户在其他小程序占用了音乐播放器时，原有小程序内的音乐将停止播放。wx.playBackgroundAudio(OBJECT)参数说明如表 3.11 所示。

表 3.11 wx.playBackgroundAudio(OBJECT)参数说明

属　性	类　型	必填	说　明
dataUrl	String	是	音乐链接
title	String	否	音乐标题
coverImgUrl	String	否	封面URL
success	Function	否	接口调用成功的回调函数
fail	Function	否	接口调用失败的回调函数
complete	Function	否	接口调用结束的回调函数（无论调用成功、失败都会执行）

示例代码如下所示。

```
Page({
  onLoad:function(){
    wx.playBackgroundAudio({
      dataUrl: 'http://y.gtimg.cn/music/photo_new/T002R300x300M000003rsKF44GyaSk.jpg?max_age=2592000',
      title:'此时此刻',
      coverImgUrl:'http://stream.qqmusic.tc.qq.com/132636799.mp3'
    })
  }
})
```

（1）使用 wx.pauseBackgroundAudio()暂停播放音乐。

（2）使用 wx.stopBackgroundAudio()停止播放音乐。

（3）使用 wx.onBackgroundAudioPlay(CALLBACK) 监听音乐播放。

（4）使用 wx.onBackgroundAudioPause(CALLBACK) 监听音乐暂停。

（5）使用 wx.onBackgroundAudioStop 监听音乐停止。

3.13　wx.seekBackgroundAudio(OBJECT)控制音乐播放进度

微信小程序使用 wx.seekBackgroundAudio(OBJECT)可以控制音乐播放的进度，设置音乐播放指定多少秒的位置，wx. seekBackgroundAudio (OBJECT)参数说明如表 3.12 所示。

表 3.12　wx. seekBackgroundAudio(OBJECT)参数说明

属　　性	类　　型	必填	说　　明
position	Number	是	音乐位置，单位：秒
success	Function	否	接口调用成功的回调函数
fail	Function	否	接口调用失败的回调函数
complete	Function	否	接口调用结束的回调函数（无论调用成功、失败都会执行）

示例代码如下所示。

```
Page({
  onLoad: function () {
    wx.seekBackgroundAudio({
      position: 30
    })
  }
})
```

3.14　wx.getBackgroundAudioPlayerState 获取音乐播放状态

微信小程序使用 wx.getBackgroundAudioPlayerState 获取后台音乐播放状态，包括选定音频的长度、选定音频的播放位置、播放状态、音频的下载进度、歌曲数据链接这五种音乐播放状态，wx.getBackgroundAudioPlayerState(OBJECT)参数说明如表 3.13 所示。

表 3.13　wx.getBackgroundAudioPlayerState(OBJECT)参数说明

属　　性	类　　型	必填	说　　明
success	Function	否	接口调用成功的回调函数
fail	Function	否	接口调用失败的回调函数
complete	Function	否	接口调用结束的回调函数（无论调用成功、失败都会执行）

要想使用 wx.getBackgroundAudioPlayerState 获取音乐播放状态，需要先使用 wx.playB

ackgroundAudio 播放音乐或者使用 wx.pauseBackgroundAudio()停止音乐播放，这样才能获取到音乐状态，示例代码如下所示。

```
<view style="display:flex;flex-direction:row;">
<button type="primary" bindtap="audioPlay" style="width:70px;height:30px;line-height:30px;background:#cccccc;">播放</button>
<button type="primary" bindtap="audioPause" style="width:70px;height:30px;line-height:30px;background:#cccccc;">暂停</button>
<button type="primary" bindtap="getState" style="width:70px;height:30px;line-height:30px;background:#cccccc;">获取状态</button>
</view>

Page({
  audioPlay: function () {
    wx.playBackgroundAudio({
      dataUrl: 'http://ws.stream.qqmusic.qq.com/M500001VfvsJ21xFqb.mp3?guid=ffffffff82def4af4b12b3cd9337d5e7&uin=346897220&vkey=6292F51E1E384E06DCBDC9AB7C49FD713D632D313AC4858BACB8DDD29067D3C601481D36E62053BF8DFEAF74C0A5CCFADD6471160CAF3E6A&fromtag=46'
    })
  },
  audioPause: function () {
      wx.pauseBackgroundAudio()
  },
  getState:function(){
     wx.getBackgroundAudioPlayerState({
      complete: function (res) {
        console.log(res);
        var status = res.status
        console.log(status);
        var dataUrl = res.dataUrl
        var currentPosition = res.currentPosition
        var duration = res.duration
        var downloadPercent = res.downloadPercent
      }
    })
  }
})
```

获取的状态值如图 3.9 所示。

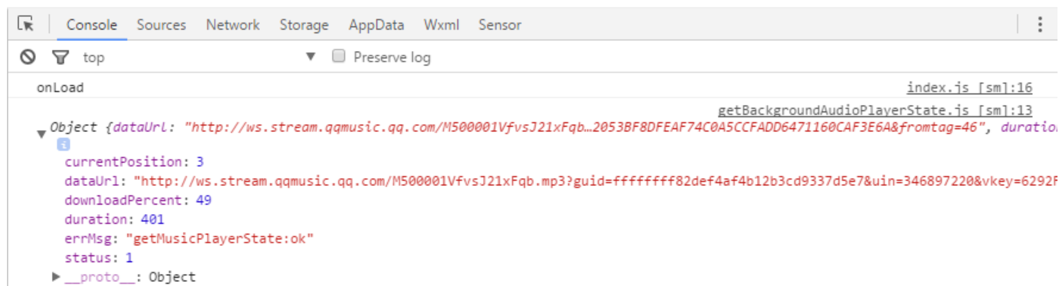


图 3.9　获取的音乐播放状态值

3.15　wx.createVideoContext(videoId)视频组件控制

视频组件控制类似于音频组件，需要创建并返回 video 上下文 videoContext 对象，videoContext 通过 videoId 与一个 video 组件绑定，通过它可以操作一个 video 组件。videoContext 对象有四个方法：play()播放、pause()暂停、seek()跳转到指定位置、sendDanmu()发送弹幕。

示例代码如下所示。

```
<view class="section tc">
  <video id="myVideo" src="http://wxsnsdy.tc.qq.com/105/20210/snsdyvideodownload?filekey=30280201010421301f0201690402534804102ca905ce620b1241b726bc41dcff44e00204012882540400&bizid=1023&hy=SH&fileparam=302c020101042530230204136ffd93020457e3c4ff02024ef202031e8d7f02030f42400204045a320a0201000400" enable-danmu danmu-btn controls></video>
  <view class="btn-area">
    <input bindblur="bindInputBlur"/>
    <button bindtap="bindSendDanmu">发送弹幕</button>
  </view>
</view>
```

```
function getRandomColor () {
  let rgb = []
  for (let i = 0 ; i < 3; ++i){
    let color = Math.floor(Math.random() * 256).toString(16)
    color = color.length == 1 ? '0' + color : color
    rgb.push(color)
  }
```

```
  return '#' + rgb.join('')
}

Page({
  onReady: function (res) {
    this.videoContext = wx.createVideoContext('myVideo')
  },
  inputValue: '',
  bindInputBlur: function(e) {
    this.inputValue = e.detail.value
  },
  bindSendDanmu: function () {
    this.videoContext.sendDanmu({
      text: this.inputValue,
      color: getRandomColor()
    })
  }
})
```

3.16　wx.chooseVideo(OBJECT)选择视频

微信小程序使用 wx.chooseVideo 从本地相册选择视频或使用相机拍摄视频，选择成功后会返回文件列表的临时路径，wx. chooseVideo (OBJECT)参数说明如表 3.14 所示。

表 3.14　wx. chooseVideo(OBJECT)参数说明

属　　性	类　　型	必填	说　　明
sourceType	StringArray	否	album 从相册选择视频，camera 使用相机拍摄，默认为：['album', 'camera']
maxDuration	Number	否	视频最长拍摄时间，单位s。最长支持60 s
camera	String	否	默认调用的为前置还是后置摄像头。front：前置，back：后置，默认 back
success	Function	否	成功则返回图片的本地文件路径列表 tempFilePaths
fail	Function	否	接口调用失败的回调函数
complete	Function	否	接口调用结束的回调函数（无论调用成功、失败都会执行）

success 返回五个参数：tempFilePath 选定视频的临时文件路径、duration 选定视频的时

间长度、size 选定视频的数据量大小、height 返回选定视频的长、width 返回选定视频的宽。文件的临时路径在小程序本次启动期间可以正常使用，如需持久保存，就要主动调用 wx.saveFile，这样在小程序下次启动时才能成功访问。

示例代码如下所示。

```
<view class="container">
    <video src="{{src}}"></video>
    <button bindtap="bindButtonTap">获取视频</button>
</view>

Page({
    bindButtonTap: function() {
        var that = this
        wx.chooseVideo({
            sourceType: ['album','camera'],
            maxDuration: 60,
      camera: 'back',
            success: function(res) {
                that.setData({
                    src: res.tempFilePath
                })
            }
        })
    }
})
```

3.17　wx.saveVideoToPhotosAlbum 保存视频到相册

微信小程序支持将视频保存到系统相册里，但前提是需要用户授权才可以保存到系统相册里，wx. saveVideoToPhotosAlbum (OBJECT)参数说明如表 3.15 所示。

表 3.15　wx. saveVideoToPhotosAlbum(OBJECT)参数说明

属　　性	类　　型	必填	说　　明
filePath	StringArray	是	图片文件路径，可以是临时文件路径也可以是永久文件路径
success	Function	否	接口调用成功的回调函数
fail	Function	否	接口调用失败的回调函数
complete	Function	否	接口调用结束的回调函数（无论调用成功、失败都会执行）

wx. saveVideoToPhotosAlbum 调用成功后会返回来一个参数：errMsg 调用结果。

示例代码如下所示。

```
Page({
  data: {
    imgUrl: ''
  },
  onLoad: function () {
    var page = this;
    wx.downloadFile({
      url: " http://wxsnsdy.tc.qq.com/105/20210/snsdyvideodownload?filekey=30280201010421
301f0201690402534804102ca905ce620b1241b726bc41dcff44e00204012882540400&bizid=102
3&hy=SH&fileparam=302c020101042530230204136ffd93020457e3c4ff02024ef202031e8d7f020
30f42400204045a320a0201000400 ",
      success: function (res) {
        console.log(res);
        var tempPath = res.tempFilePath;

        wx. saveVideoToPhotosAlbum ({
          filePath: tempPath,
          success:function(res){
            console.log(res);
          }
        })
      }
    })

  }
})
```

3.18　wx.saveFile 保存文件

微信小程序可以将文件的临时路径保存为本地路径，这样下次启动微信小程序的时候，就可以像访问本地文件那样访问到这些文件，而这就需要使用 wx.saveFile 保存文件这个接口，wx.saveFile(OBJECT)参数说明如表 3.16 所示。

表 3.16　wx.saveFile(OBJECT)参数说明

属　　性	类　　型	必填	说　　明
filePath	StringArray	是	图片文件路径，可以是临时文件路径，也可以是永久文件路径
success	Function	否	接口调用成功的回调函数
fail	Function	否	接口调用失败的回调函数
complete	Function	否	接口调用结束的回调函数（无论调用成功、失败都会执行）

接口调用成功后会返回 savedFilePath 文件保存的路径，本地文件存储的大小限制为 10MB。

示例代码如下所示。

```
Page({
  onLoad:function(){
    wx.chooseImage({
      success: function (res) {
        var tempFilePaths = res.tempFilePaths
        wx.saveFile({
          tempFilePath: tempFilePaths[0],
          success: function (res) {
            var savedFilePath = res.savedFilePath
          }
        })
      }
    })
  }
})
```

3.19　wx.getSavedFileList 获取本地文件列表

微信小程序可以使用 wx.saveFile 将文件保存到本地，保存到本地的文件可以通过它提供的 wx.getSavedFileList 来获取保存到本地的文件列表，wx. getSavedFileList (OBJECT)参数说明如表 3.17 所示。

表 3.17　getSavedFileList(OBJECT)参数

属　　性	类　　型	必填	说　　明
success	Function	否	接口调用成功的回调函数
fail	Function	否	接口调用失败的回调函数
complete	Function	否	接口调用结束的回调函数（无论调用成功、失败都会执行）

接口调用成功后会返回 fileList 文件列表，列表里包括 filePath 文件的本地路径、createTime 文件保存时的时间戳（从 1970/01/01 08:00:00 到当前时间的秒数）、size 文件大小（单位：B）。

示例代码如下所示。

```
Page({
  saveImg:function(){
    wx.chooseImage({
      success: function (res) {
        var tempFilePaths = res.tempFilePaths
        wx.saveFile({
          tempFilePath: tempFilePaths[0],
          success: function (res) {
            var savedFilePath = res.savedFilePath
          }
        })
      }
    })
  },
  getFile:function(){
    wx.getSavedFileList({
      success: function (res) {
        console.log(res.fileList)
      }
    })
  }
})
```

文件列表信息如图 3.10 所示。

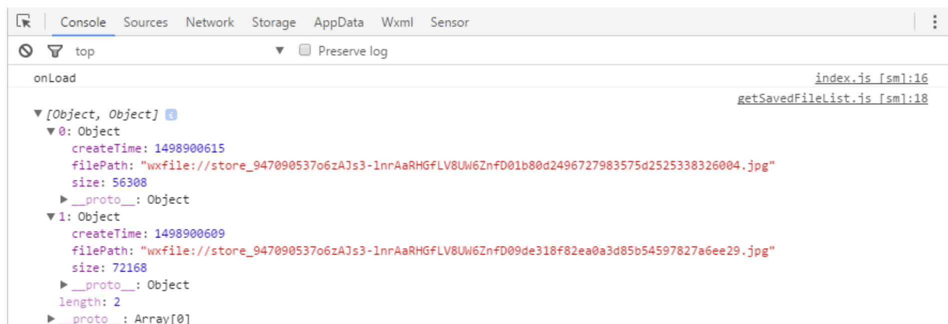


图 3.10　文件列表信息

3.20　wx. getSavedFileInfo 获取本地文件信息

微信小程序可以使用 wx. getSavedFileInfo 获取本地指定文件的信息，包括文件的大小以及创建时间，wx. getSavedFileInfo (OBJECT)参数说明如表 3.18 所示。

表 3.18　wx.getSavedFileInfo(OBJECT)参数说明

属　　性	类　型	必填	说　　明
filePath	Function	是	文件路径
success	Function	否	接口调用成功的回调函数
fail	Function	否	接口调用失败的回调函数
complete	Function	否	接口调用结束的回调函数（无论调用成功、失败都会执行）

接口调用成功后会返回文件的大小以及创建时间（文件保存时的时间戳，从 1970/01/01 08:00:00 到当前时间的秒数）。

示例代码如下所示。

```
Page({
  saveImg:function(){
    wx.chooseImage({
      success: function (res) {
        var tempFilePaths = res.tempFilePaths
        wx.saveFile({
          tempFilePath: tempFilePaths[0],
          success: function (res) {
            var savedFilePath = res.savedFilePath
```

```
            }
          })
        }
      })
    },
    getFile:function(){
      wx.getSavedFileList({
        success: function (res) {
          console.log(res.fileList)
          var fileList = res.fileList;
          for(var i=0;i<fileList.length;i++){
            var filePath = fileList[i].filePath;
            wx.getSavedFileInfo({
              filePath: filePath,
              success:function(res){
                 console.log("文件大小="+res.size+",创建时间="+res.createTime);
              }
            })

          }
        }
      })
    }
})
```

本地文件信息如图 3.11 所示。

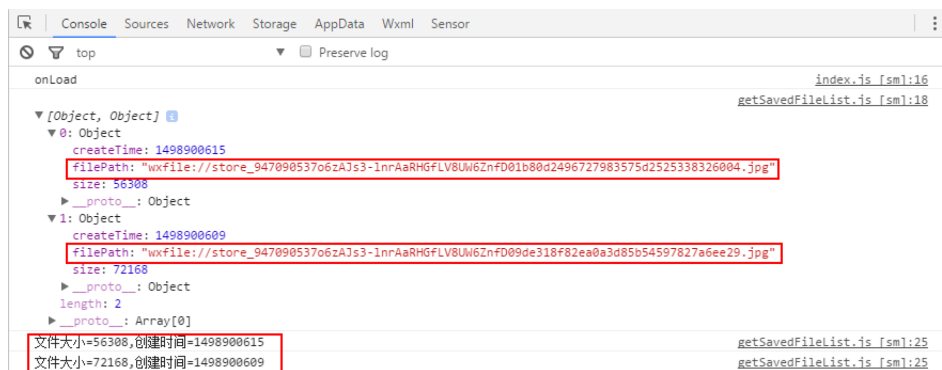


图 3.11　本地文件信息

3.21　wx. removeSavedFile 删除本地文件

微信小程序可以将文件存放到本地，同样也可以将本地文件删除，使用 wx. removeSavedFile 就可以将本地文件删除，wx. removeSavedFile (OBJECT)参数说明如表 3.19 所示。

表 3.19　wx. removeSavedFile(OBJECT)参数说明

属　　性	类　　型	必填	说　　明
filePath	Function	是	需要删除的文件路径
success	Function	否	接口调用成功的回调函数
fail	Function	否	接口调用失败的回调函数
complete	Function	否	接口调用结束的回调函数（无论调用成功、失败都会执行）

示例代码如下所示。

```
Page({
  saveImg:function(){
    wx.chooseImage({
      success: function (res) {
        var tempFilePaths = res.tempFilePaths
        wx.saveFile({
          tempFilePath: tempFilePaths[0],
          success: function (res) {
            var savedFilePath = res.savedFilePath
          }
        })
      }
    })
  },
  getFile:function(){
    wx.getSavedFileList({
      success: function (res) {
        console.log(res.fileList)
        var fileList = res.fileList;
        for(var i=0;i<fileList.length;i++){
          var filePath = fileList[i].filePath;
          wx.getSavedFileInfo({
            filePath: filePath,
```

```
                    success:function(res){
                        console.log("文件大小="+res.size+",创建时间="+res.createTime);
                    }
                })

            }
        }
    })
  },
  deleteFile:function(){
    wx.getSavedFileList({
      success: function (res) {
        if (res.fileList.length > 0) {
          wx.removeSavedFile({
            filePath: res.fileList[0].filePath,
            complete: function (res) {
              console.log(res)
            }
          })
        }
      }
    })
  }
})
```

3.22 wx. openDocument 打开文档

微信小程序支持打开文档，可以使用 wx. openDocument 打开文档，支持格式：doc、xls、ppt、pdf、docx、xlsx 和 pptx wx。openDocument (OBJECT)参数说明如表 3.20 所示。

表 3.20 wx. openDocument(OBJECT)参数说明

属　　性	类　　型	必填	说　　明
filePath	Function	是	文件路径，可通过downFile获得
success	Function	否	接口调用成功的回调函数
fail	Function	否	接口调用失败的回调函数
complete	Function	否	接口调用结束的回调函数（无论调用成功、失败都会执行）

示例代码如下所示。

```
wx.downloadFile({
  url: 'http://example.com/somefile.pdf',
  success: function (res) {
    var filePath = res.tempFilePath
    wx.openDocument({
      filePath: filePath,
      success: function (res) {
        console.log('打开文档成功')
      }
    })
  }
})
```

3.23　wx.setStorage 和 wx.setStorageSync 本地数据缓存

微信小程序提供了两个 API 接口，用于将数据缓存到本地，它们分别为 wx.setStorage 异步缓存数据到本地和 wx.setStorageSync 同步缓存数据到本地。它们可以将数据存储在本地缓存中指定的 key 中，覆盖原来该 key 对应的内容，同一个小程序本地数据缓存上限为 10MB。

wx.setStorage(OBJECT)参数说明如表 3.21 所示。

表 3.21　wx.setStorage(OBJECT)参数说明

属　　性	类　　型	必填	说　　明
key	String	是	本地缓存中的指定的 key
data	Object/String	是	需要存储的内容
success	Function	否	接口调用成功的回调函数
fail	Function	否	接口调用失败的回调函数
complete	Function	否	接口调用结束的回调函数（无论调用成功、失败都会执行）

wx.setStorageSync(KEY,DATA)参数说明如表 3.22 所示。

表 3.22　wx.setStorageSync(KEY,DATA)参数说明

属　　性	类　　型	必填	说　　明
key	String	是	本地缓存中的指定的 key
data	Object/String	是	需要存储的内容

下面有一个场景，是将学生成绩的信息缓存，如表 3.23 所示。

表 3.23 学生成绩

序号	姓名	语文	数学	英语
1	Tom	98	89	77
2	Kevin	99	87	97
3	David	80	74	65

异步缓存 wx.setStorage 和同步缓存 wx.setStorageSync 示例代码如下所示。

```
Page({
  onLoad:function(){
    //异步缓存数据
     wx.setStorage({
       key: '高一',
       data: this.loadStudents(),
     });

     //同步缓存数据
     wx.setStorageSync('高二', this.loadStudents())
  },
  loadStudents:function(){
    var Students = new Array();
    var Student = new Object();
    Student.id = '1';
    Student.name = 'Tom';
    Student.Chinese = '98';
    Student.Math =   '89';
    Student.English = '77';
    Students[0] = Student;

    var Student1 = new Object();
    Student1.id = '1';
    Student1.name = 'Kevin';
    Student1.Chinese = '99';
    Student1.Math = '87';
    Student1.English = '97';
    Students[1] = Student1;

    var Student2 = new Object();
    Student2.id = '1';
```

```
        Student2.name = 'David';
        Student2.Chinese = '80';
        Student2.Math = '74';
        Student2.English = '65';
        Students[2] = Student2;

        return Students;
    }
})
```

在 Storage 面板里可以看到缓存到本地的数据，如图 3.12 所示。

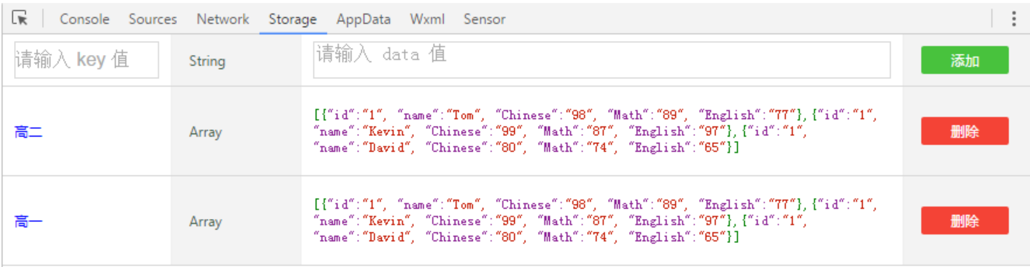


图 3.12　本地缓存数据

3.24　wx.getStorage 和 wx.getStorageSync 获取本地数据缓存

微信小程序使用 wx.setStorage 和 wx.setStorageSync 缓存数据到本地，数据缓存到本地后可以使用 wx.getStorage 异步获取本地数据缓存和 wx.getStorageSync 同步获取本地数据缓存，它们都是从本地指定的 key 中对应的内容。

wx.getStorage(OBJECT)参数说明如表 3.24 所示。

表 3.24　wx. getStorage(OBJECT)参数说明

属　　性	类　　型	必填	说　　明
key	String	是	本地缓存中的指定的 key
success	Function	否	接口调用成功的回调函数，res = {data: key对应的内容}
fail	Function	否	接口调用失败的回调函数
complete	Function	否	接口调用结束的回调函数（无论调用成功、失败都会执行）

wx.getStorage(OBJECT)返回一个参数 data，就是 key 对应的内容。

wx.getStorageSync(KEY)参数就是本地缓存中指定的 key，返回指定 key 对应的内容。

在 3.23 节中通过 wx.setStorage 异步和 wx.setStorageSync 同步的方式将学生成绩缓存到本地，下面通过 wx.getStorage 异步和 wx.getStorageSync 同步的方式获取本地数据，示例代码如下所示。

```
Page({
  onLoad:function(){
      //异步获取本地缓存数据
      wx.getStorage({
        key: '高一',
        success: function (res) {
          console.log(res.data)
        }
      });
      //同步获取本地缓存数据
      try {
        var value = wx.getStorageSync('高二');
        console.log(value);
        if (value) {
          // Do something with return value
        }
      } catch (e) {
        // Do something when catch error
      }
  }

})
```

通过 key 值就可以获取相应的内容，如图 3.13 所示。



图 3.13　获取本地数据

3.25　wx.getStorageInfo 和 wx.getStorageInfoSync 获取缓存信息

微信小程序的 wx.setStorage 异步和 wx.setStorageSync 同步获取本地数据，都是通过指定的 key 值才能获取对应的内容，如果想获取多个 key 对应的内容，就可以使用

wx.getStorageInfo 异步和 wx.getStorageInfoSync 同步的方式获取本地缓存数据的 key 值集合，获取到 key 值集合后再使用 wx.setStorage 异步和 wx.setStorageSync 同步的方式获取具体的数据信息。

wx.getStorageInfo 异步和 wx.getStorageInfoSync 同步方式可以获取本地 keys 当前 storage 中所有的 key、currentSize 当前占用的空间大小（单位 kb）、limitSize 限制的空间大小（单位 kb），示例代码如下所示。

```
Page({
  onLoad:function(){
    //异步获取缓存信息
    wx.getStorageInfo({
      success: function(res) {
        console.log(res.keys);
        console.log(res.currentSize);
        console.log(res.limitSize);
      }
    })
    //同步获取缓存信息
    try {
      var res = wx.getStorageInfoSync()
      console.log(res.keys)
      console.log(res.currentSize)
      console.log(res.limitSize)
    } catch (e) {
      // Do something when catch error
    }
  }
})
```

数据缓存信息如图 3.14 所示。

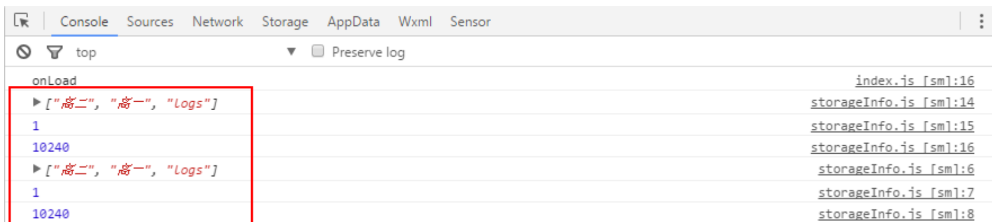


图 3.14　数据缓存信息

3.26　清理本地缓存数据

清理本地缓存数据分为清理指定 key 对应的数据和清理所有数据，使用 wx.removeStorage 异步和 wx.removeStorageSync 同步的方式移除指定 key 对应的内容，使用 wx.clearStorage()异步和 wx.clearStorageSync()同步的方式清理所有本地缓存数据。

wx.removeStorage(OBJECT)参数说明如表 3.25 所示。

表 3.25　wx.removeStorage(OBJECT)参数说明

属　　性	类　　型	必填	说　　明
key	String	是	本地缓存中的指定的 key
success	Function	否	接口调用成功的回调函数
fail	Function	否	接口调用失败的回调函数
complete	Function	否	接口调用结束的回调函数（无论调用成功、失败都会执行）

wx.removeStorageSync(KEY)只有一个参数，就是指定的 key 值。

wx.clearStorage()和 wx.clearStorageSync()都是没有参数的。

示例代码如下所示。

```
Page({
  onLoad:function(){
     //异步移除指定 key 的值
    wx.removeStorage({
      key: '高一',
      success: function (res) {
        console.log(res.data)
      }
    });
    //同步移除指定 key 的值
    try {
      wx.removeStorageSync('key')
    } catch (e) {
      // Do something when catch error
    }

    //异步清理本地所有缓存数据
    wx.clearStorage();
    //同步清理本地所有缓存数据
    try {
```

```
        wx.clearStorageSync()
      } catch (e) {
        // Do something when catch error
      }
  }
})
```

3.27 wx.getLocation 获取位置

获取坐标位置是经常会用到的功能，例如，使用天气 App 获取天气情况、使用美团 App 进行团购、使用地图查找路线等这些场景下都需要获取坐标位置，微信小程序也提供了获取坐标位置（wx.getLocation）的接口，通过这个接口可以获取当前位置的经度和纬度。

wx.getLocation(OBJECT)参数说明如表 3.26 所示。

表 3.26 wx.getLocation(OBJECT)参数说明

属　性	类　型	必填	说　明
type	String	否	默认为 wgs84 返回 gps 坐标，gcj02 返回可用于wx.openLocation 的坐标
success	Function	是	接口调用成功的回调函数，详见返回参数说明
fail	Function	否	接口调用失败的回调函数
complete	Function	否	接口调用结束的回调函数（调用成功、失败都会执行）

接口调用成功后返回的参数如表 3.27 所示。

表 3.27 success返回的参数

参　数	说　明
latitude	纬度，浮点数，范围为-90~90，负数表示南纬
longitude	经度，浮点数，范围为-180~180，负数表示西经
speed	速度，浮点数，单位m/s
accuracy	位置的精确度
altitude	高度，单位 m
verticalAccuracy	垂直精度，单位 m（Android 无法获取，返回 0）
horizontalAccuracy	水平精度，单位 m

下面通过获取当前坐标位置，查看天气情况，如图 3.15 所示。



图 3.15　天气情况

（1）在 getLocation.wxml 页面文件里，进行今天天气情况的布局设计，具体代码如下所示。

```
<view  class="content">
    <view  class="today">
        <view  class="info">
                <view  class="temp">{{today.wendu}}˚C</view>
                <view  class="lowhigh">{{today.todayInfo.low}}/{{today.todayInfo.high}}</view>
                <view  class="type">{{today.todayInfo.type}}</view>
                <view  class="city">{{city}}</view>
                <view  class="week">{{today.todayInfo.date}}</view>
                <view  class="weather"> {{today.todayInfo.fengxiang}}  {{today.todayInfo.fengli}}</vi
ew>

        </view>
    </view>
</view>
```

（2）在 getLocation.wxss 样式文件里，给今天天气情况的布局添加样式，具体代码如下所示。

```
.content{
     font-family ： 微软雅黑,宋体;
     font-size:  14px;
     background-size:cover;
     height:  100%;
     width:100%;
     background:#4781C1;
     color:#FFFFFF;
```

```
}



.today{
    padding-top:70rpx;
    height:50%;
}

.temp{
    font-size:80px;
    text-align: center;
}

.city{
    color: #ffffff;
    font-size: 20px;
    text-align: center;
    margin-top:20rpx;
    margin-right: 10rpx;
}
.lowhigh{
    color:#ffffff;
    font-size: 12px;
    text-align: center;
    margin-top: 30rpx;
}
.type{
    color:#ffffff;
    font-size: 16px;
    text-align: center;
    margin-top: 30rpx;
}
.week{
    color:#ffffff;
    font-size: 12px;
    text-align: center;
    margin-top: 30rpx;
}
```

```
.weather{
    color:#ffffff;
    font-size: 12px;
    text-align: center;
    margin-top: 20rpx;
}
```

（3）在 getLocation.js 业务逻辑处理文件里，先获取当前位置，再获取天气情况，具体代码如下所示。

```
Page({
  data: {
    city: "",
    today: {}
  },
  onLoad: function () {
    this.loadInfo();
  },
  loadInfo: function () {
    var page = this;
    wx.getLocation({
      type: 'gcj02', // 默认为 wgs84 返回 GPS 坐标，gcj02 返回可用于 wx.openLocation 的坐标
      success: function (res) {
        var latitude = res.latitude;
        var longitude = res.longitude;
        page.loadCity(latitude, longitude);
      }

    })
  },
  loadCity: function (latitude, longitude) {
    var page = this;
    wx.request({
      url: 'http://api.map.baidu.com/geocoder/v2/',
      data: {
        ak: 'D6WOzHaymzVVKvgiy8UbhQEznkgeK6BD',
        location: latitude + ',' + longitude,
        output: 'json'
      },
      method: 'GET', // OPTIONS, GET, HEAD, POST, PUT, DELETE, TRACE, CONNECT
      header: {
        'Content-Type': 'application/json'
```

```
        },
        success: function (res) {
          console.log(res);
          var city = res.data.result.addressComponent.city;
          console.log(city);
          city = city.replace("市", "");
          page.setData({ city: city });
          page.loadWeather(city);
        }

      })
    },
    loadWeather: function (city) {
      var page = this;
      wx.request({
        url: 'http://wthrcdn.etouch.cn/weather_mini',
        data: {
          city: city
        },
        method: 'GET', // OPTIONS, GET, HEAD, POST, PUT, DELETE, TRACE, CONNECT
        header: {
          'Content-Type': 'application/json'
        },
        success: function (res) {
          console.log(res.data.data);
          console.log(res.data.data.forecast);
          var today = res.data.data;
          var future = today.forecast;
          var todayInfo = future.shift();
          var high = todayInfo.high;
          var low = todayInfo.low;
          todayInfo.high = high.substring(2, high.length);
          todayInfo.low = low.substring(2, low.length);
          console.log(todayInfo);
          today.todayInfo = todayInfo;
          page.setData({ today: today });
        }

      })
    }
})
```

3.28　wx. chooseLocation 选择位置

微信小程序使用 wx.chooseLocation 打开地图来选择位置，它可以返回 latitude 纬度、longitude 经度、name 位置名称以及 address 详细地址，wx.chooseLocation(OBJECT)参数说明如表 3.28 所示。

表 3.28　wx.chooseLocation(OBJECT)参数说明

属　　性	类　　型	必填	说　　明
success	Function	否	接口调用的回调函数，详见返回参数说明
cancel	Function	否	用户取消时调用
fail	Function	否	接口调用失败的回调函数
complete	Function	否	接口调用结束的回调函数（无论调用成功、失败都会执行）

使用 wx.chooseLocation 接口会弹出地图供我们选择位置，如图 3.16 和图 3.17 所示。



图 3.16　位置地图



图 3.17　搜索位置

示例代码如下所示。

```
Page({
  onLoad:function(){
```

```
        wx.chooseLocation({
          success: function(res){
              console.log(res);
          }
        })
    }
})
```

3.29 wx.openLocation 打开内置地图

微信小程序内置了腾讯地图，可以使用 wx.openLocation(OBJECT)这个接口打开地图，它和 wx. chooseLocation 的区别是，wx. chooseLocation 是在地图上或者用搜索位置的方式来选择位置，而 wx.openLocation 是通过指定的纬度和经度来确定位置。wx.openLocation(OBJECT)的参数说明如表 3.29 所示。

表 3.29 wx.openLocation(OBJECT)参数说明

属　性	类　型	必填	说　明
latitude	Float	是	纬度，范围为-90~90，负数表示南纬
longitude	Float	是	经度，范围为-180~180，负数表示西经
scale	INT	否	缩放比例，范围5~18，默认为18
name	String	否	位置名
address	String	否	地址的详细说明
success	Function	否	接口调用成功的回调函数
fail	Function	否	接口调用失败的回调函数
complete	Function	否	接口调用结束的回调函数（无论调用成功、失败都会执行）

示例代码如下所示。

```
Page({
  onLoad:function(){
    wx.getLocation({
        type: 'gcj02', //返回可用于 wx.openLocation 的经纬度
        success: function(res) {
          var latitude = res.latitude
          var longitude = res.longitude
          wx.openLocation({
            latitude: latitude,
```

```
                    longitude: longitude,
                    scale: 28,
                    })
              }
          })
   }
})
```

界面效果如图 3.18 所示。



图 3.18 打开位置

3.30 获取设备系统信息

微信小程序针对获取设备系统信息提供了两个 API 接口：一个是 wx.getSystemInfo 异步获取设备系统信息，另一个是 wx.getSystemInfoSync 同步获取设备系统信息，它们的返回参数是一样的，如表 3.30 所示。

表 3.30 返回参数说明

参 数	说 明
model	手机型号
pixelRatio	设备像素比
screenWidth	屏幕宽度
screenHeight	屏幕高度
windowWidth	窗口宽度
windowHeight	窗口高度
language	微信设置的语言
version	微信版本号
system	操作系统版本
platform	客户端平台
SDKVersion	客户端基础库版本

示例代码如下所示。

```
Page({
  onLoad:function(){
    //异步获取设备系统信息
    wx.getSystemInfo({
      success: function (res) {
        console.log("=================== 异 步 获 取 设 备 系 统 信 息
======================");
        console.log("手机型号:"+res.model)
        console.log("设备像素比:" +res.pixelRatio)
        console.log("屏幕宽度:" + res.screenWidth)
        console.log("屏幕高度:" + res.screenHeight)
        console.log("可使用窗口宽度:" +res.windowWidth)
        console.log("可使用窗口高度:" +res.windowHeight)
        console.log("微信设置的语言:" +res.language)
        console.log("微信版本号:" +res.version)
        console.log("操作系统版本:" + res.system)
        console.log("客户端平台:" +res.platform)
        console.log("客户端基础库版本:" + res.SDKVersion)
      }
    });

   //同步获取设备系统信息
    try {
```

```
        var res = wx.getSystemInfoSync()
        console.log("======================= 同步获取设备系统信息======================");
        console.log("手机型号:" + res.model)
        console.log("设备像素比:" + res.pixelRatio)
        console.log("屏幕宽度:" + res.screenWidth)
        console.log("屏幕高度:" + res.screenHeight)
        console.log("可使用窗口宽度:" + res.windowWidth)
        console.log("可使用窗口高度:" + res.windowHeight)
        console.log("微信设置的语言:" + res.language)
        console.log("微信版本号:" + res.version)
        console.log("操作系统版本:" + res.system)
        console.log("客户端平台:" + res.platform)
        console.log("客户端基础库版本:" + res.SDKVersion)
      } catch (e) {
        // Do something when catch error
      }

   }
})
```

设备系统信息如图 3.19 所示。

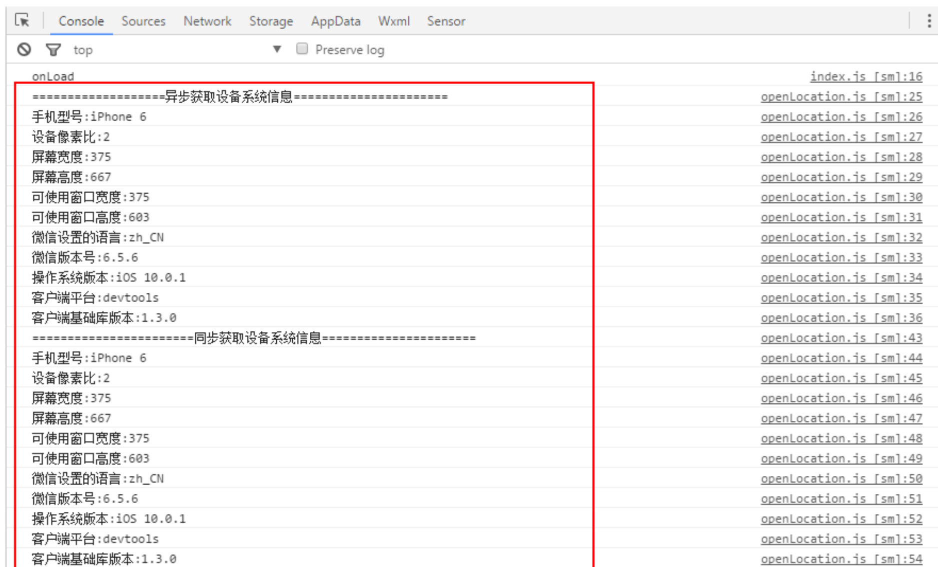


图 3.19　设备系统信息

3.31　获取网络状态

微信小程序提供了获取网络类型 wx.getNetworkType 和监听网络状态变化 wx.onNetworkStatusChange(CALLBACK)这两个接口。wx.getNetworkType 返回一个参数，就是网络的类型 networkType。wx.onNetworkStatusChange 返回两个参数：isConnected 当前是否有网络连接和 networkType 网络类型。networkType 网络类型有 wifi（WiFi 网络）、2g（2G 网络）、3g（3G 网络）、4g（4G 网络）、none（无网络）和 unknown（Android 下不常见的网络类型）。

示例代码如下所示。

```
Page({
  onLoad:function(){
    //获取网络类型
    wx.getNetworkType({
      success: function (res) {
        // 返回网络类型，有效值：
        // wifi/2g/3g/4g/unknown(Android 下不常见的网络类型)/none(无网络)
        var networkType = res.networkType
      }
    });
    //监听网络状态变化
    wx.onNetworkStatusChange(function (res) {
      console.log(res.isConnected)
      console.log(res.networkType)
    })
  }
})
```

3.32　wx.onAccelerometerChange 加速度计

微信小程序可以监听加速度数据，频率：5 次/s，接口调用后会自动开始监听，可以使用 wx.stopAccelerometer 停止监听。最终返回 *X* 轴、*Y* 轴、*Z* 轴的坐标值，示例代码如下所示。

```
Page({
  onLoad:function(){
```

```
        //监听加速度数据
          wx.startAccelerometer()//开始监听
          wx.onAccelerometerChange(function (res) {
            console.log("x:" + res.x)
            console.log("y:" + res.y)
            console.log("z" + res.z)
          })
          wx.stopAccelerometer()//停止监听
  }
})
```

3.33　wx.onCompassChange 监听罗盘数据

微信小程序可以监听罗盘数据，频率：5 次/s，接口调用后会自动开始监听，可以使用 wx.stopCompass 停止监听。示例代码如下所示。

```
Page({
  onLoad:function(){
      //监听加速度数据
      wx.startCompass()//开始监听
      wx.onCompassChange(function (res) {
            console.log(res.direction)
      })
      wx.stopCompass()//停止监听
  }
})
```

3.34　wx.makePhoneCall 拨打电话和 wx.scanCode 扫码

在微信小程序里支持拨打电话功能，使用 wx.makePhoneCall 这个 API 就可以拨打电话，wx. makePhoneCall (OBJECT)的参数有四个属性，如表 3.31 所示。

表 3.31　wx. makePhoneCall (OBJECT)参数说明

属　　性	类　　型	必填	说　　明
phoneNumber	String	是	需要拨打的电话号码

续表

属　　性	类　　型	必填	说　　明
Success	Function	否	接口调用成功的回调
fail	Function	否	接口调用失败的回调函数
complete	Function	否	接口调用结束的回调函数（无论调用成功、失败都会执行）

示例代码如下所示。

```
onlyFromCamerawx.makePhoneCall({
  phoneNumber: '1340000' //仅为示例，并非真实的电话号码
})
```

wx.scanCode(OBJECT) 调起客户端扫码界面，扫码成功后返回对应的结果，它的参数有四个属性，如表 3.32 所示。

表 3.32　wx、scanCode(OBJECT)参数说明

属　　性	类　　型	必填	说　　明
onlyFromCamera	Boolean	否	是否只能从相机扫码，不允许从相册选择图片
success	Function	否	接口调用成功的回调函数，返回内容详见返回参数说明
fail	Function	否	接口调用失败的回调函数
complete	Function	否	接口调用结束的回调函数（无论调用成功、失败都会执行）

接口调用成功后返回四个参数：result（扫码的内容）、scanType（扫码的类型）、charSet（扫码的字符集）和 path（当所扫的码为当前小程序的合法二维码时，会返回此字段，内容为二维码携带的 path）。

示例代码如下所示。

```
// 允许从相机和相册扫码
wx.scanCode({
  success: (res) => {
    console.log(res)
  }
})

// 只允许从相机扫码
wx.scanCode({
  onlyFromCamera: true,
  success: (res) => {
    console.log(res)
  }
})
```

3.35　剪贴板和屏幕亮度

微信小程序可以设置指定内容到剪贴板上，然后再从剪贴板获取内容，只需使用 wx.setClipboardData 设置剪贴板内容，再使用 wx.getClipboardData 获取剪贴板内容即可。wx.setClipboardData(OBJECT)参数说明如表 3.33 所示。

表 3.33　wx.setClipboardData(OBJECT)参数说明

属　　性	类　　型	必填	说　　明
data	String	是	需要设置的内容
success	Function	否	接口调用成功的回调函数，返回内容详见返回参数说明
fail	Function	否	接口调用失败的回调函数
complete	Function	否	接口调用结束的回调函数（无论调用成功、失败都会执行）

wx.getClipboardData(OBJECT)有一个返回值 data，就是剪贴板的内容，这样就获取了剪贴板内容。

示例代码如下所示。

```
Page({
  onLoad:function(){
    wx.setClipboardData({//设置剪贴板内容
      data: '我是剪贴板内容',
      success:function(res){
        wx.getClipboardData({ //获取剪贴板内容
          success: function (res) {
            console.log(res.data) //打印剪贴板内容
          }
        })
      }
    })
  }
})
```

微信小程序还可以设置屏幕的亮度以及获取屏幕的亮度，通过 wx.setScreenBrightness 设置屏幕的亮度，通过 wx.getScreenBrightness 获取屏幕的亮度。wx.setScreenBrightness (OBJECT)设置屏幕亮度的参数如表 3.34 所示。

表 3.34 wx. setScreenBrightness(OBJECT)参数说明

属　性	类　型	必填	说　明
value	Number	是	屏幕亮度值，范围0~1，0最暗，1最亮
success	Function	否	接口调用成功的回调函数，返回内容详见返回参数说明
fail	Function	否	接口调用失败的回调函数
complete	Function	否	接口调用结束的回调函数（无论调用成功、失败都会执行）

wx.getScreenBrightness(OBJECT)有一个返回值 value，就是屏幕的亮度值，范围 0~1，0 最暗，1 最亮。

示例代码如下所示。

```
Page({
  onLoad: function () {
    wx.setScreenBrightness({//设置屏幕亮度值
      value: 1,
      success: function (res) {
        wx.getScreenBrightness({ //获取屏幕亮度值
          success: function (res) {
            console.log(res.value) //打印屏幕亮度值
          }
        })
      }
    })
  }
})
```

3.36 wx.addPhoneContact 添加联系人

微信小程序可以通过“新增联系人”或“添加到已有联系人”的方式来添加联系人，并把联系人写入手机系统通讯录，完成手机通讯录联系人和联系方式的添加。使用 wx.addPhoneContact 这个接口可以完成联系人的添加，wx.addPhoneContact(OBJECT)的参数说明如表 3.35 所示。

表 3.35 wx. addPhoneContact (OBJECT)参数说明

属　性	类　型	必填	说　明
photoFilePath	String	否	头像本地文件路径

续表

属　　性	类　型	必填	说　　明
nickName	String	否	昵称
lastName	String	否	姓氏
middleName	String	否	中间名
firstName	String	是	名字
remark	String	否	备注
mobilePhoneNumber	String	否	手机号
weChatNumber	String	否	微信号
addressCountry	String	否	联系地址国家
addressState	String	否	联系地址省份
addressCity	String	否	联系地址城市
addressStreet	String	否	联系地址街道
addressPostalCode	String	否	联系地址邮政编码
organization	String	否	公司
title	String	否	职位
workFaxNumber	String	否	工作传真
workPhoneNumber	String	否	工作电话
hostNumber	String	否	公司电话
email	String	否	电子邮件
url	String	否	网站
workAddressCountry	String	否	工作地址国家
workAddressState	String	否	工作地址省份
workAddressCity	String	否	工作地址城市
workAddressStreet	String	否	工作地址街道
workAddressPostalCode	String	否	工作地址邮政编码
homeFaxNumber	String	否	住宅传真
homePhoneNumber	String	否	住宅电话
homeAddressCountry	String	否	住宅地址国家
homeAddressState	String	否	住宅地址省份
homeAddressCity	String	否	住宅地址城市
homeAddressStreet	String	否	住宅地址街道
homeAddressPostalCode	String	否	住宅地址邮政编码
success	Function	否	接口调用成功的回调函数，返回内容详见返回参数说明
fail	Function	否	接口调用失败的回调函数
complete	Function	否	接口调用结束的回调函数（无论调用成功、失败都会执行）

回调结果如表 3.36 所示。

表 3.36　回调结果

回调类型	errMsg	说　　明
success	ok	添加成功
fail	fail cancel	用户取消操作
fail	fail ${detail}	调用失败，detail 加上详细信息

示例代码如下所示。

```
Page({
  onLoad:function(){
    wx.addPhoneContact({
      firstName: '名字',
      nickName:'昵称',
      lastName:'姓氏',
      mobilePhoneNumber:'手机号'
    })
  }
})
```

3.37　wx.showToast 和 wx.showLoading 显示提示框

微信小程序提供了两种提示框：一种是 wx.showToast 消息提示框，另外一种是 wx.showLoading 显示 loading 提示框。需主动调用 wx.hideLoading 才能关闭提示框，如图 3.20 和图 3.21 所示。

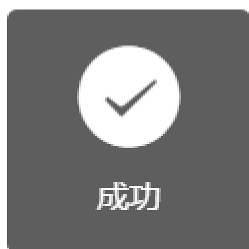


图 3.20　消息提示框



图 3.21　loading 提示框

wx.showToast(OBJECT)显示消息提示框，它的参数说明如表 3.37 所示。

表 3.37　wx.showToast(OBJECT)显示消息提示框参数说明

属　　性	类　　型	必填	说　　明
title	String	是	提示的内容
icon	String	否	图标，只支持“success”、“loading”
image	String	否	自定义图标的本地路径，image 的优先级高于 icon
duration	Number	否	提示的延迟时间，单位毫秒，默认为1500，最大为10000
mask	Boolean	否	是否显示透明蒙层，防止触摸穿透，默认为false
success	Function	否	接口调用成功的回调函数
fail	Function	否	接口调用失败的回调函数
complete	Function	否	接口调用结束的回调函数（无论调用成功、失败都会执行）

wx.showLoading(OBJECT)显示 loading 提示框的参数说明如表 3.38 所示。

表 3.38　wx.showLoading(OBJECT)显示loading提示框参数说明

属　　性	类　　型	必填	说　　明
title	String	是	提示的内容
mask	Boolean	否	是否显示透明蒙层，防止触摸穿透，默认为false
success	Function	否	接口调用成功的回调函数
fail	Function	否	接口调用失败的回调函数
complete	Function	否	接口调用结束的回调函数（无论调用成功、失败都会执行）

使用 wx.hideToast()隐藏消息提示框和 wx.hideLoading()隐藏 loading 提示框。

示例代码如下所示。

（1）在 toast.wxml 页面文件里，定义了四个按钮：显示消息提示框、隐藏消息提示框、显示 loading 提示框以及隐藏 loading 提示框，具体代码如下所示。

```
<view style="margin:10px;text-align:center;">
<button type="primary" size="mini" bindtap="toast" style="width:160px;margin-right:10px;">showToast 消息提示框</button>
<button type="primary" size="mini"  bindtap="loading" style="width:160px;margin-right:10px;">loading 提示框</button>
<button type="primary" size="mini"  bindtap="hideToast" style="width:160px;margin-right:10px;">隐藏消息提示框</button>
<button type="primary" size="mini"  bindtap="hideLoading" style="width:160px;margin-right:10px;">隐藏 loading 提示框</button>
</view>
```

（2）在 toast.js 业务逻辑处理文件里，分别给四个按钮添加相应事件，具体代码如下所示。

```
Page({
  toast:function(){
    wx.showToast({//显示消息提示框
      title: '成功',
      icon: 'success',
      duration: 2000
    })
  },
  loading:function(){
    wx.showLoading({//显示 loading 提示框
      title: '加载中',
    })
    setTimeout(function () {
      wx.hideLoading()
    }, 2000)
  },
  hideToast:function(){//隐藏消息提示框
    wx.hideToast();
  },
  hideLoading:function(){//隐藏 loading 提示框
    wx.hideLoading();
  }
})
```

3.38 wx.showModal 显示模态弹窗

微信小程序使用 wx.showModal 来显示模态弹窗，模态弹窗会覆盖底下的内容，从而使用户只能操作提供的“确定”和“取消”按钮，避免用户的误操作，如图 3.22 所示。



图 3.22　模态弹窗

wx.showModal(OBJECT)显示模态弹窗参数说明如表 3.39 所示。

表 3.39　wx.showModal(OBJECT)参数说明

属　　性	类　　型	必填	说　　明
title	String	是	提示的标题
content	String	是	提示的内容
showCancel	Boolean	否	是否显示取消按钮，默认为 true
cancelText	String	否	取消按钮的文字，默认为“取消”，最多 4 个字符
cancelColor	HexColor	否	取消按钮的文字颜色，默认为“#000000”
confirmText	String	否	确定按钮的文字，默认为“确定”，最多 4 个字符
confirmColor	HexColor	否	确定按钮的文字颜色，默认为“#3CC51F”
success	Function	否	接口调用成功的回调函数，返回res.confirm为true时，表示用户单击确定按钮
fail	Function	否	接口调用失败的回调函数
complete	Function	否	接口调用结束的回调函数（无论调用成功、失败都会执行）

示例代码如下所示。

```
Page({
  onLoad:function(){
    wx.showModal({
```

```
            title: '温馨提示',
            content: '模态弹窗可以点击确定或者取消！',
            success: function (res) {
              if (res.confirm) {
                console.log('您点击的是确定按钮')
              } else if (res.cancel) {
                console.log('您点击的是取消按钮')
              }
            }
          })
    }
})
```

3.39 wx.showActionSheet 显示操作菜单

微信小程序使用 wx.showActionSheet 这个 API 接口来显示操作菜单，从底部逐渐滑向顶部，并提供了多个菜单选项供选择，每次只能选择一项，或者单击“取消”按钮，一项也不选择，如图 3.23 所示。



图 3.23 操作菜单

wx.showActionSheet(OBJECT)参数说明如表 3.40 所示。

表 3.40　wx.showActionSheet(OBJECT)参数说明

属　　性	类　　型	必填	说　　明
itemList	String Array	是	按钮的文字数组，数组长度最大为6个
itemColor	HexColor	否	按钮的文字颜色，默认为“#000000”
success	Function	否	接口调用成功的回调函数，详见返回参数说明
fail	Function	否	接口调用失败的回调函数
complete	Function	否	接口调用结束的回调函数（无论调用成功、失败都会执行）

示例代码如下所示。

```
Page({
  onLoad: function () {
    var edu = ['中专', '大专', '本科', '硕士', '博士'];
    wx.showActionSheet({
      itemList: edu,
      success: function (res) {
        if (!res.cancel) {
          console.log(res.tapIndex);//数组序号
          console.log(edu[res.tapIndex]);//数组序号对应的值
        }
      }
    })
  }
})
```

3.40　设置导航条

微信小程序提供了设置导航条标题的 API 接口，可以使用 wx.setNavigationBarTitle 动态设置当前页面的标题，wx.setNavigationBarTitle(OBJECT)参数说明如表 3.41 所示。

表 3.41　wx.setNavigationBarTitle(OBJECT)参数说明

属　　性	类　　型	必填	说　　明
title	String	是	页面标题
success	Function	否	接口调用成功的回调函数，详见返回参数说明
fail	Function	否	接口调用失败的回调函数
complete	Function	否	接口调用结束的回调函数（无论调用成功、失败都会执行）

不仅可以使用 wx.setNavigationBarTitle 来设置导航标题，同时还可以使用 wx.showNavigationBarLoading()在当前页面显示导航条加载动画和 wx.hideNavigationBarLoading()隐藏导航条加载动画，如图 3.24 所示。

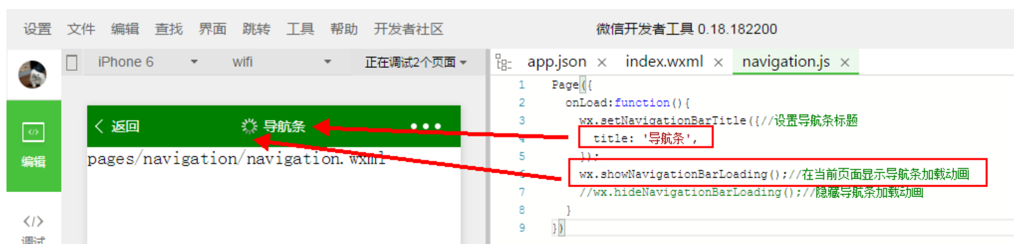


图 3.24　设置导航条

示例代码如下所示。

```
Page({
  onLoad:function(){
    wx.setNavigationBarTitle({//设置导航条标题
      title: '导航条',
    });
    wx.showNavigationBarLoading();//在当前页面显示导航条加载动画
    //wx.hideNavigationBarLoading();//隐藏导航条加载动画
  }
})
```

3.41　wx.createAnimation 创建动画

微信小程序可以创建动画效果，提供了 wx.createAnimation 这个 API 接口用来创建动画效果，它会创建一个动画实例 animation，然后调用实例的方法来描述动画，最后通过动画实例的 export 方法导出动画数据传递给组件的 animation 属性。wx.createAnimation (OBJECT)参数说明如表 3.42 所示。

表 3.42　wx.createAnimation (OBJECT)参数说明

属　　性	类　型	必填	默 认 值	说　　明
duration	Integer	否	400	动画持续时间，单位ms
timingFunction	String	否	"linear"	定义动画的效果
delay	Integer	否	0	动画延迟时间，单位ms

续表

属　　性	类　型	必填	默 认 值	说　　明
transformOrigin	Function	否	"50% 50% 0"	设置transform-origin

在 wx.createAnimation(OBJECT)参数说明里有定义动画效果的属性 timingFunction，它的有效值如下。

（1）linear：动画从头到尾的速度是相同的。

（2）ease：动画以低速开始，然后加快，在结束前变慢。

（3）ease-in：动画以低速开始。

（4）ease-in-out：动画以低速开始和结束。

（5）ease-out：动画以低速结束。

（6）step-start：动画第一帧就跳至结束状态直到结束。

（7）step-end：动画一直保持开始状态，最后一帧跳到结束状态。

示例代码如下所示。

```
<view   style="margin-top: 150px;margin-left:130px;"  bindtap="search">
<image  style="position: absolute;width: 100px;height: 100px;"  src="../../images/normal.png"></i
mage>
<image   style="position: absolute;width: 100px;height: 100px;"   animation="{{searchAnimation}}
"  src="../../images/searching.png"></image>
  </view>

Page({
  data: {
    searchAnimation: {},//放大动画
  },
  search:function(){
    var page = this;
    var animation = wx.createAnimation({//ch 创建动画实例
      duration: 1000
    })
    animation.opacity(0).scale(3, 3).step();//修改透明度,放大
    this.setData({
      searchAnimation: animation.export() //导出动画数据
    })
  }
})
```

动画效果如图 3.25 和图 3.26 所示。

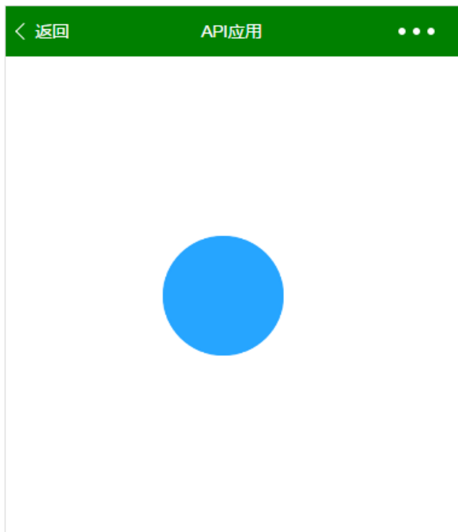


图 3.25　搜索前

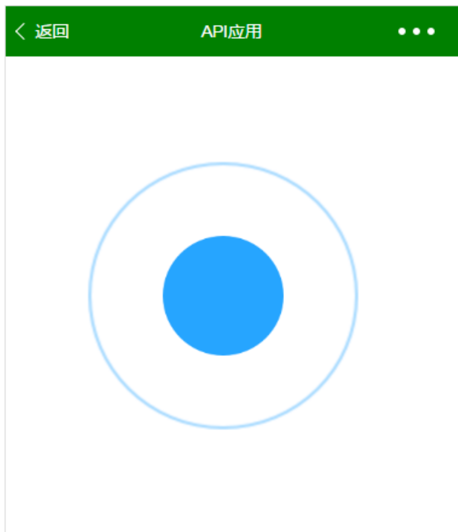


图 3.26　搜索后

animation 动画实例提供了很多动画效果，动画效果调用结束后会返回自身，支持链式调用的写法，动画实例方法如表 3.43 所示。

表 3.43　animation动画实例方法

方法类型	方　　法	参　　数	说　　明
样式	opacity	value	透明度，参数范围 0~1
	backgroundColor	color	颜色值
	width	length	长度值，如果传入 Number，则默认使用 px，可传入其他自定义单位的长度值
	height	length	长度值，如果传入 Number，则默认使用 px，可传入其他自定义单位的长度值
	top	length	长度值，如果传入 Number，则默认使用 px，可传入其他自定义单位的长度值
	left	length	长度值，如果传入 Number，则默认使用 px，可传入其他自定义单位的长度值
	bottom	length	长度值，如果传入 Number，则默认使用 px，可传入其他自定义单位的长度值
	right	length	长度值，如果传入 Number，则默认使用 px，可传入其他自定义单位的长度值

续表

方法类型	方　　法	参　　数	说　　明
旋转	rotate	deg	deg的范围-180~180，从原点顺时针旋转一个deg角度
	rotateX	deg	deg的范围-180~180，在*X*轴旋转一个deg角度
	rotateY	deg	deg的范围-180~180，在*Y*轴旋转一个deg角度
	otateZ	deg	deg的范围-180~180，在*Z*轴旋转一个deg角度
	rotate3d	(x,y,z,deg)	同transform-function rotate3d
缩放	scale	sx,[sy]	一个参数时，表示在*X*轴、*Y*轴同时缩放sx倍数；两个参数时表示在*X*轴缩放sx倍数，在*Y*轴缩放sy倍数
	scaleX	sx	在*X*轴缩放sx倍数
	scaleY	sy	在*Y*轴缩放sy倍数
	scaleZ	sz	在*Z*轴缩放sy倍数
	scale3d	(sx,sy,sz)	在*X*轴缩放sx倍数，在*Y*轴缩放sy倍数，在*Z*轴缩放sz倍数
偏移	translate	tx,[ty]	一个参数时，表示在*X*轴偏移tx，单位px；两个参数时，表示在*X*轴偏移tx，在*Y*轴偏移ty，单位px
	translateX	tx	在*X*轴偏移tx，单位px
	translateY	ty	在*Y*轴偏移tx，单位px
	translateZ	tz	在*Z*轴偏移tx，单位px
	translate3d	(tx,ty,tz)	在*X*轴偏移tx，在*Y*轴偏移ty，在*Z*轴偏移tz，单位px
倾斜	skew	ax,[ay]	参数范围-180~180；一个参数时，*Y*轴坐标不变，*X*轴坐标延顺时针倾斜ax度；两个参数时，分别在*X*轴倾斜ax度，在*Y*轴倾斜ay度
	skewX	ax	参数范围-180~180；*Y*轴坐标不变，*X*轴坐标延顺时针倾斜ax度
	skewY	ay	参数范围-180~180；*X*轴坐标不变，*Y*轴坐标延顺时针倾斜ay度
矩阵变形	matrix	(a,b,c,d,tx,ty)	同transform-function matrix
	matrix3d		同transform-function matrix3d

animation 不仅可以调用一次动画效果，还可以调用一组动画效果。当调用动画操作方法后可调用 step() 来表示一组动画完成，可以在一组动画中调用任意多个动画方法，一组动画中的所有动画会同时开始，一组动画完成后才会进行下一组动画。step 可以传入一个与 wx.createAnimation() 一样的配置参数用于指定当前组动画的配置，如图 3.27 和图 3.28 所示。

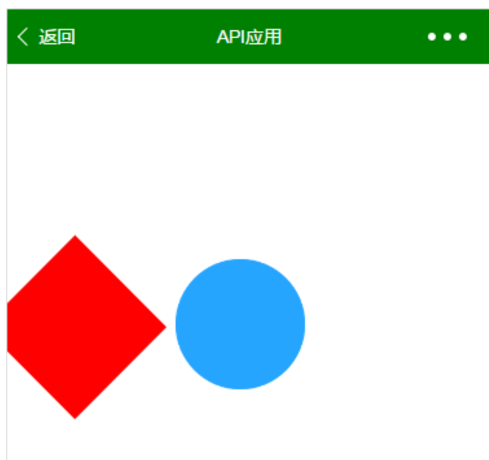


图 3.27　动作一

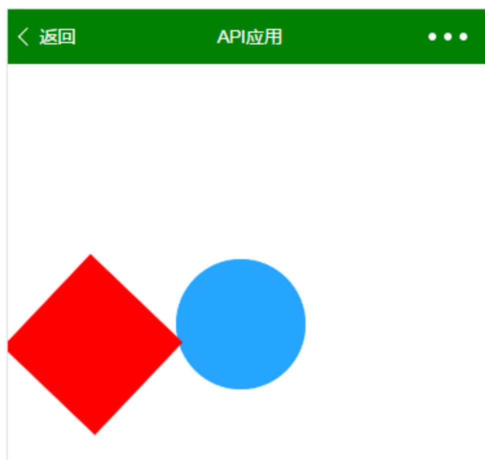


图 3.28　动作二

示例代码如下所示。

```
<view  style="margin-top: 150px;margin-left:130px;" bindtap="search">
<image style="position: absolute;width: 100px;height: 100px;" src="../../images/normal.png"></i
mage>
<image  style="position: absolute;width: 100px;height: 100px;"  animation="{{searchAnimation}}
" src="../../images/searching.png"></image>
  </view>
<view animation="{{animationData}}" style="background:red;height:100rpx;width:100rpx"></vie
w>


Page({
  data: {
    searchAnimation: {},//放大动画
    animationData: {}
  },
  search: function () {
    var page = this;
    var animation = wx.createAnimation({//ch 创建动画实例
      duration: 5000
    })
    animation.opacity(0).scale(3, 3).step();//修改透明度,放大
    this.setData({
      searchAnimation: animation.export() //导出动画数据
    })
```

```
},
onShow: function () {
  var animation = wx.createAnimation({
    duration: 5000,
    timingFunction: 'ease',
  })

  this.animation = animation

  animation.scale(2, 2).rotate(45).step()

  this.setData({
    animationData: animation.export()
  })

  setTimeout(function () {
    animation.translate(30).step()
    this.setData({
      animationData: animation.export()
    })
  }.bind(this), 1000)
},
rotateAndScale: function () {
  // 旋转同时放大
  this.animation.rotate(45).scale(2, 2).step()
  this.setData({
    animationData: this.animation.export()
  })
},
rotateThenScale: function () {
  // 先旋转后放大
  this.animation.rotate(45).step()
  this.animation.scale(2, 2).step()
  this.setData({
    animationData: this.animation.export()
  })
},
rotateAndScaleThenTranslate: function () {
  // 先旋转同时放大，然后平移
  this.animation.rotate(45).scale(2, 2).step()
```

```
    this.animation.translate(100, 100).step({ duration: 1000 })
    this.setData({
      animationData: this.animation.export()
    })
  }
})
```

3.42 获取第三方平台数据

微信小程序提供了两种获取第三方平台数据的方式：一种是 wx.getExtConfig(OBJECT)异步方式获取，另外一种是 wx.getExtConfigSync()同步方式获取。wx.getExtConfig(OBJECT)异步获取第三方平台数据的参数说明如表 3.44 所示。

表 3.44 wx.getExtConfig(OBJECT)参数说明

属　性	类　型	必填	说　明
success	Function	否	接口调用成功的回调函数，详见返回参数说明
fail	Function	否	接口调用失败的回调函数
complete	Function	否	接口调用结束的回调函数（无论调用成功、失败都会执行）

wx.getExtConfig(OBJECT)异步获取第三方平台数据的返回值有两个：errMsg 调用结果和 extConfig 第三方平台自定义的数据，示例代码如下所示。

```
if(wx.getExtConfig) {
  wx.getExtConfig({
    success: function (res) {
      console.log(res.extConfig)
    }
  })
}
```

wx.getExtConfigSync()同步获取第三方平台数据的返回值只有一个，即 extConfig 第三方平台自定义的数据，示例代码如下所示。

```
var extConfig = wx.getExtConfigSync? wx.getExtConfigSync(): {}
console.log(extConfig)
```

3.43　wx.login 登录

微信小程序的登录可以分为六个步骤：

（1）微信小程序调用 wx.login 登录接口获取登录凭证（code）。

（2）微信小程序将登录凭证（code）传输给后台服务器。

（3）后台服务器用登录凭证（code）来获取 session_key 和 openid。

（4）获取到 session_key 和 openid 之后，代表微信小程序登录是合法有效的，后台服务器接着创建登录状态的 session，传输给微信小程序。

（5）微信小程序获取从后台传输过来的登录状态的 session 后，通过 wx.setStorage 保存到本地。

（6）微信小程序调用 wx.checkSession 接口检测当前用户登录态是否有效，登录态过期后开发者可以再次调用 wx.login 获取新的用户登录状态。

1．获取登录凭证（code）

微信小程序调用 wx.login 调用接口获取登录凭证（code），然后将登录凭证传输给后台服务器，用来换取用户登录状态信息，包括用户的唯一标识（openid）以及本次登录的会话密钥（session_key）。

示例代码如下所示。

```
App({
  onLaunch: function() {
    wx.login({//获取登录凭证 code
      success: function(res) {
        if (res.code) {
          //发起网络请求,将 code 值传输给后台服务器
          wx.request({
            url: 'https://test.com/onLogin',
            data: {
              code: res.code
            }
          })
        } else {
          console.log('获取用户登录态失败！ ' + res.errMsg)
        }
      }
```

```
        });
    }
})
```

2．根据登录凭证（code）获取 ession_key 和 openid

根据登录凭证（code）获取 ession_key 和 openid，这是一个 HTTPS 接口，开发者服务器使用登录凭证 code 获取 session_key 和 openid。其中，session_key 是对用户数据进行加密签名的密钥，为了自身应用安全，session_key 不应在网络上传输。接口地址如下所示。

```
https://api.weixin.qq.com/sns/jscode2session?appid=APPID&secret=SECRET&js_code=JSCODE
&grant_type=authorization_code
```

接口请求参数：appid 是小程序唯一标识，secret 是小程序的 app secret，js_code 是登录时获取的 code，grant_type 应填写为 authorization_code。

返回值如下所示。

```
//正常返回的 JSON 数据包
{
      "openid": "OPENID",
      "session_key": "SESSIONKEY"
}
//错误时返回的 JSON 数据包（示例为 Code 无效）
{
    "errcode": 40029,
    "errmsg": "invalid code"
}
```

3．wx.checkSession(OBJECT)登录状态维护

用户登录态拥有一定的时效性。用户越是长时间未使用小程序，用户登录状态越有可能失效。开发者只需调用 wx.checkSession 接口检测当前用户登录状态是否有效即可。登录关态过期后，开发者可以再次调用 wx.login 来获取新的用户登录状态。

```
wx.checkSession({
  success: function(){
    //session 未过期，并且在本生命周期一直有效
  },
  fail: function(){
    //登录状态过期
    wx.login() //重新登录
    ....
  }
})
```

3.44　wx.authorize 授权

在使用微信小程序的时候，小程序的部分接口需要获得用户同意后才能调用。调用此类接口时，如果用户未授权，则会弹窗询问用户，用户单击“同意”后方可调用接口。如果用户单击了“拒绝”，则短期内调用时不再出现弹窗，而是直接进入 fail 回调，如图 3.29 所示。



图 3.29　授权

用户可以在小程序设置界面中修改对该小程序的授权信息。本接口用于提前向用户发起授权，调用后会立刻弹窗询问用户是否同意小程序使用某项功能或获取用户的某些数据，但不会实际调用接口。如果用户之前已经同意，则不再出现弹窗，而是直接返回成功。

wx.authorize(OBJECT)授权的参数说明如表 3.45 所示。

表 3.45 wx.authorize (OBJECT)参数说明

属　性	类　型	必填	说　明
scope	String	是	需要获取权限的scope
success	Function	否	接口调用成功的回调函数，详见返回参数说明
fail	Function	否	接口调用失败的回调函数
complete	Function	否	接口调用结束的回调函数（无论调用成功、失败都会执行）

调用 wx.authorize (OBJECT)授权接口后会有一个返回值 errMsg 调用结果，获取权限的 scope 列表如表 3.46 所示。

表 3.46 scope权限列表

Scope	对应接口	说　明
scope.userInfo	wx.getUserInfo	用户信息
scope.userLocation	wx.getLocation, wx.chooseLocation	地理位置
scope.address	wx.chooseAddress	通信地址
scope.record	wx.startRecord	录音功能
scope.writePhotosAlbum	wx.saveImageToPhotosAlbum, wx.saveVideoToPhotosAlbum	保存到相册

假如想调用用户的地理位置，则示例代码如下所示。

```
Page({
  onLoad:function(){
    wx.getSetting({
      success(res) {
        if (!res['scope.userLocation']) {
          wx.authorize({
            scope: 'scope.userLocation',
            success() {
              // 用户已经同意小程序使用获取位置功能，后续调用 wx.getLocation 接口不会弹窗
询问
              wx.getLocation({
                success: function(res) {
                  console.log(res);
                },
              })
            }
          })
        }
```

```
        }
      })
    }
})
```

3.45　wx.getUserInfo 用户信息

微信小程序在登录之后，就可以获取用户的信息，也就是说，微信小程序需要先调用 wx.login 接口，然后才能获取用户信息。获取用户信息使用 wx.getUserInfo 用户信息接口，wx.getUserInfo(OBJECT)参数说明如表 3.47 所示。

表 3.47　wx.getUserInfo(OBJECT)参数说明

属性	类型	必填	说明
ithCredentials	Boolean	否	是否带上登录状态信息
success	Function	否	接口调用成功的回调函数，详见返回参数说明
fail	Function	否	接口调用失败的回调函数
complete	Function	否	接口调用结束的回调函数（无论调用成功、失败都会执行）

调用 wx.getUserInfo(OBJECT)接口后会有五个返回值。

（1）userInfo：用户信息对象，不包含 openid 等敏感信息。

（2）rawData ：不包含敏感信息的原始数据字符串，用于计算签名。

（3）signature：使用 sha1(rawData + sessionkey) 得到字符串，用于校验用户信息，参考文档 signature。

（4）encryptedData：包括敏感数据在内的完整用户信息的加密数据，详细内容见加密数据解密算法；

（5）iv：加密算法的初始向量，详细内容见加密数据解密算法。

示例代码如下所示。

```
//app.js
App({
  onLaunch: function() {
    //调用 API 从本地缓存中获取数据
    var logs = wx.getStorageSync('logs') || []
    logs.unshift(Date.now())
    wx.setStorageSync('logs', logs)
  },
```

```
  getUserInfo: function(cb) {
    var that = this
    if (this.globalData.userInfo) {
      typeof cb == "function" && cb(this.globalData.userInfo)
    } else {
      //调用登录接口
      wx.getUserInfo({
        withCredentials: false,
        success: function(res) {
          var userInfo = res.userInfo
          var nickName = userInfo.nickName
          var avatarUrl = userInfo.avatarUrl
          var gender = userInfo.gender //性别 0：未知，1：男，2：女
          var province = userInfo.province
          var city = userInfo.city
          var country = userInfo.country
          that.globalData.userInfo = res.userInfo
          typeof cb == "function" && cb(that.globalData.userInfo)
        }
      })
    }
  },

  globalData: {
    userInfo: null
  }
})
```

encryptedData 解密后为以下 JSON 结构，示例代码如下所示。

```
{
    "openId": "OPENID",
    "nickName": "NICKNAME",
    "gender": GENDER,
    "city": "CITY",
    "province": "PROVINCE",
    "country": "COUNTRY",
    "avatarUrl": "AVATARURL",
    "unionId": "UNIONID",
    "watermark":
    {
```

```
            "appid":"APPID",
        "timestamp":TIMESTAMP
        }
}
```

3.46　wwx.requestPayment 微信支付

微信小程序提供了微信支付的接口 wx.requestPayment 用来进行微信支付，这个 requestPayment 接口确实是小程序前端唯一需要调用的函数，但是微信支付的成功接入实际远远不止这一个函数。

1．微信支付准备工作

（1）申请微信支付。微信小程序通过认证后，就可以申请微信支付了，微信支付申请后，会发送微信支付商户号、商户平台用户名、密码等信息到注册者邮箱。

（2）配置小程序密钥，微信公众平台的开发设置里，会随机生成 AppSecret。这个 AppSecret 用于和微信服务器交互，比如获取用户的 openid 接口时就需要用到。

（3）设置密钥和下载证书，用申请微信支付获得的用户名和密码，登录商户平台（pay.weixin.qq.com），在账户中心的 API 安全处下载证书和设置密钥。

注意：密钥是 32 位，设置以后需要妥善保管，所有微信支付相关的接口都会使用这个密钥加密。

（4）配置 HTTPS 服务器，微信小程序属于前端框架开发，后台服务器是开发者自己的服务器，小程序发起的是 HTTPS 请求，开发者必须配置 HTTPS 服务器，配置服务器先要获取证书，证书可以向相关机构购买，腾讯云目前可以向用户提供免费的证书。

2．微信支付流程工作

（1）支付前工作，微信小程序通过接口将支付的信息（订单号、价格等）发送给开发者服务器，开发者服务器拿到这些信息后提交给微信服务器，返回需要支付的一些信息，然后传输给微信小程序，示例代码如下所示。

```
{
        "sign":"A2***A6",
        "timestamp":"14***68",
        "package":"Sign=WXPay",
        "partnerId":"14***02",
        "appid":"wx***ab",
        "nonceStr":"9f***37",
```

```
    "prepayId":"wx***54"
}
```

（2）生成 paySign 签名，wx.requestPayment(OBJECT)支付需要 paySign 签名参数，https://pay.weixin.qq.com/wiki/doc/api/wxa/wxa_api.php?chapter=7_3&index=1&t=20161122，微信小程序 MD5 加密工具下载地址：https://code.csdn.net/snippets/2019875/master/download。

（3）发起支付工作，微信小程序从开发服务器获得一些支付信息后，通过调用 wx.requestPayment(OBJECT)开始进行支付工作，wx.requestPayment(OBJECT)参数说明如表 3.48 所示。

表 3.48 wx.requestPayment(OBJECT)参数说明

属　　性	类　型	必填	说　　明
timeStamp	String	是	时间戳从 1970 年 1 月 1 日 00:00:00 至今的秒数，即当前的时间
nonceStr	String	是	随机字符串，长度为32个字符以下
package	String	是	统一下单接口返回的 prepay_id 参数值，提交格式如：prepay_id=*
signType	String	是	签名算法，暂支持 MD5
paySign	String	是	签名，具体签名方案参见小程序支付接口文档https://pay.weixin.qq.com/wiki/doc/api/wxa/wxa_api.php?chapter=10_4#
success	Function	否	接口调用成功的回调函数，详见返回参数说明
fail	Function	否	接口调用失败的回调函数
complete	Function	否	接口调用结束的回调函数（无论调用成功、失败都会执行）

示例代码如下所示。

```
wx.requestPayment({
    'timeStamp': '',
    'nonceStr': '',
    'package': '',
    'signType': 'MD5',
    'paySign': '',
    'success':function(res){
    },
    'fail':function(res){
    }
})
```

wx.requestPayment(OBJECT)回调结果如表 3.49 所示。

表 3.49　回调结果

回调类型	errMsg	说　　明
success	requestPayment:ok	调用支付成功
fail	requestPayment:fail cancel	用户取消支付
fail	requestPayment:fail (detail message)	支付失败，其中 detail message 为后台返回的详细失败原因

3.47　模板消息

模板消息是用户本人在微信体系内与页面有交互行为后触发的，可以给用户推送服务通知，如推送购买成功通知、订单发货提醒、订单支付成功通知、交易提醒等模板消息。

模板消息使用分为三个步骤。

（1）获取模板消息，登录 https://mp.weixin.qq.com 微信公众平台获取模板，如果没有合适的模板，则可以申请添加新模板，审核通过后即可使用，如图 3.30 所示。

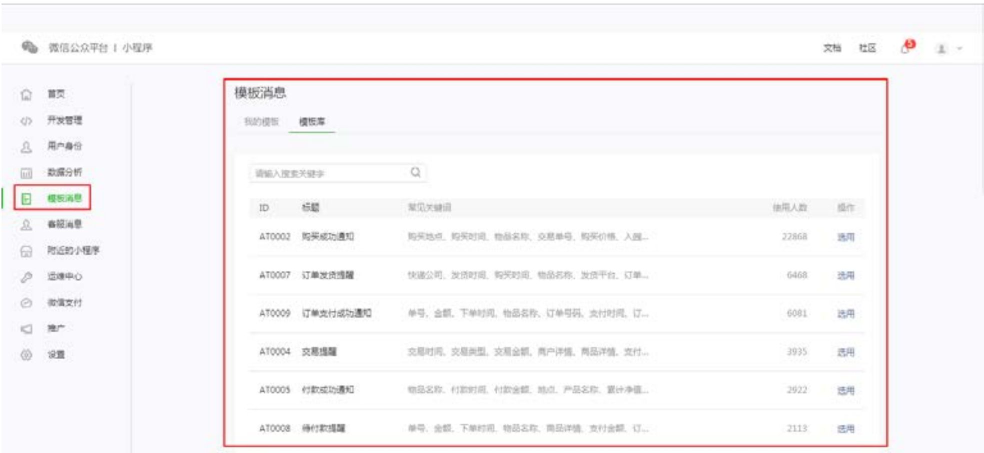


图 3.30　模板库

（2）使用页面的 <form/> 组件，设置属性 report-submit 为 true 时，可以声明为需要发送模板消息，此时单击按钮提交表单可以获取 formId，用于发送模板消息。或者当用户完成支付行为后，可以获取 prepay_id 用于发送模板消息。

（3）获取 access_token，access_token 是全局唯一接口调用凭据，开发者调用各接口时都需要使用 access_token。access_token 的存储至少要保留 512 个字符空间。access_token 的

有效期目前为 2 个小时，需要定时刷新，重复获取将导致上次获取的 access_token 失效。

接口地址：

```
https://api.weixin.qq.com/cgi-bin/token?grant_type=client_credential&appid=APPID&secret=APPS
ECRET
```

它需要传递三个参数。

（1）grant_type：获取 access_token 填写 client_credential。

（2）appid：第三方用户唯一凭证。

（3）secret：第三方用户唯一凭证密钥，即 appsecret。

返回值有两个。

（1）ccess_token：获取的凭证。

（2）expires_in：凭证有效时间，单位：秒。

返回值示例代码如下所示：

```
{"access_token": "ACCESS_TOKEN", "expires_in": 7200}
```

（3）发送模板消息，获取 access_token 后，就可以发送模板消息了，发送模板消息的接口如下所示：

```
https://api.weixin.qq.com/cgi-bin/message/wxopen/template/send?access_token=ACCESS_TOK
EN
```

POST 传递参数如表 3.50 所示。

表 3.50　POST传递参数

参　　数	必填	说　　明
touser	是	接收者（用户）的 openid
template	是	所需下发模板消息的id
page	否	单击模板卡片后的跳转页面，仅限本小程序内的页面。支持带参数（示例index?foo=bar）。该字段不填写则模板无跳转
form_id	是	表单提交场景下，为 submit 事件带上的 formId；支付场景下，为本次支付的 prepay_id
data	是	模板内容，不填则下发空模板
color	否	模板内容字体的颜色，不填则默认黑色
emphasis_keyword	否	模板需要放大的关键词，不填则默认无放大

示例代码如下所示。

```
{
  "touser": "OPENID",
  "template_id": "TEMPLATE_ID",
```

```
  "page": "index",
  "form_id": "FORMID",
  "data": {
        "keyword1": {
              "value": "339208499",
              "color": "#173177"
        },
        "keyword2": {
              "value": "2015 年 01 月 05 日  12:30",
              "color": "#173177"
        },
        "keyword3": {
              "value": "粤海喜来登酒店",
              "color": "#173177"
        } ,
        "keyword4": {
              "value": "广州市天河区天河路 208 号",
              "color": "#173177"
        }
  },
  "emphasis_keyword": "keyword1.DATA"
}
```

在调用完发送模板消息后会返回 JSON 数据，如果模板消息发送正常，则返回错误码 errcode，示例代码如下所示。

```
{
  "errcode": 0,
  "errmsg": "ok",
}
```

当发送模板消息失败时，会返回 errcode 错误码。

（1）40037：template_id 不正确。

（2）41028：form_id 不正确，或者过期。

（3）41029：form_id 已被使用。

（4）41030：page 不正确。

（5）45009：接口调用超过限额（目前默认每个账号日调用限额为 100 万）。

3.48　转发信息

微信小程序支持将当前页面转发出去，分享给好友或者好友群，这时就可以调用 onShareAppMessage(options)接口来转发消息，它是在 Page 中定义 onShareAppMessage 函数，设置该页面的转发信息。只有定义了此事件的处理函数，右上角菜单才会显示“转发”按钮，用户单击转发按钮的时候会调用，此事件需要 return 一个 Object，用于自定义转发内容，onShareAppMessage(options)参数说明如表 3.51 所示。

表 3.51　onShareAppMessage(options)参数说明

参　数	必填	说　明
from	String	转发事件来源。button：页面内转发按钮；menu：右上角转发菜单
target	Object	如果from值是 button，则 target 是触发这次转发事件的 button，否则为 undefined

转发字段包括 title 转发标题（当前小程序名称），path 转发路径（当前页面 path）。必须是以 / 开头的完整路径，转发成功后会返回值 shareTickets，即 shareTicket 数组，每一项是一个 shareTicket ，对应一个转发对象，示例代码如下所示。

```
Page({
  onShareAppMessage: function (res) {
    if (res.from === 'button') {
      // 来自页面内的转发按钮
      console.log(res.target)
    }
    return {
      title: '自定义转发标题',
      path: '/page/user?id=123',
      success: function(res) {
        // 转发成功
      },
      fail: function(res) {
        // 转发失败
      }
    }
  }
})
```

微信小程序还提供了 wx.showShareMenu(OBJECT) 显示当前页面的转发按钮、wx.hideShareMenu(OBJECT) 隐藏转发按钮、wx.updateShareMenu(OBJECT) 更新转发属

性、wx.getShareInfo(OBJECT) 获取转发详细信息，示例代码如下所示。

```
//显示当前页面的转发按钮
wx.showShareMenu({
  withShareTicket: true
})

//隐藏转发按钮
wx.hideShareMenu();

//更新转发属性
wx.updateShareMenu({
  withShareTicket: true,
  success() {
  }
})
```

3.49　wx.chooseAddress 收货地址

微信小程序提供了 wx.chooseAddress(OBJECT)收货地址 API，它可以调取用户编辑收货地址原生界面，并在编辑完成后返回用户选择的地址，接口需要用户授权才可以使用。wx.chooseAddress(OBJECT)参数说明如表 3.52 所示。

表 3.52　wx.chooseAddress(OBJECT)参数说明

属　性	类　型	必填	说　明
success	Function	否	返回用户选择的收货地址信息
fail	Function	否	接口调用失败的回调函数
complete	Function	否	接口调用结束的回调函数（无论调用成功、失败都会执行）

wx.chooseAddress(OBJECT)接口调用成功后，返回值如下所示。

（1）errMsg：调用结果。

（2）userName：收货人姓名。

（3）postalCode：邮编。

（4）provinceName：国标收货地址第一级地址。

（5）cityName：国标收货地址第二级地址。

（6）countyName：国标收货地址第三级地址。

（7）detailInfo：详细收货地址信息。

（8）nationalCode ：收货地址国家码。

（9）telNumber：收货人手机号码。

示例代码如下所示。

```
wx.chooseAddress({
  success: function (res) {
    console.log(res.userName)
    console.log(res.postalCode)
    console.log(res.provinceName)
    console.log(res.cityName)
    console.log(res.countyName)
    console.log(res.detailInfo)
    console.log(res.nationalCode)
    console.log(res.telNumber)
  }
})
```

3.50 卡券

微信小程序针对卡券提供了两个 API 接口：一个是 wx.addCard(OBJECT)，批量添加卡券；另一个是 wx.openCard(OBJECT)，查看微信卡包中的卡券。调用 wx.addCard(OBJECT) 可以批量添加卡券，wx.addCard(OBJECT)参数说明如表 3.53 所示。

表 3.53 wx.chooseAddress(OBJECT)参数说明

属 性	类 型	必填	说 明
cardList	ObjectArray	是	需要添加的卡券列表，列表内对象说明请参见请求对象说明
success	Function	否	接口调用成功的回调函数
fail	Function	否	接口调用失败的回调函数
complete	Function	否	接口调用结束的回调函数（无论调用成功、失败都会执行）

cardList（卡券列表）是一个对象数组，它是卡券的一个集合，卡券信息包含 cardId（卡券 Id）和 cardExt（卡券的扩展参数）。cardExt（卡券的扩展参数）需 JSON 序列化为字符串后方可传入。

卡券的扩展参数说明如表 3.54 所示。

表 3.54　cardExt卡券的扩展参数说明

属　　性	类　型	必填	是否参与签名	说　　明
code	String	否	是	用户领取的code，仅自定义code模式的卡券才能填写，非自定义code模式卡券不能填写
openid	String	否	是	指定领取者的openid，只有该用户能领取。bind_openid字段为true的卡券必须填写，为false的可不填写
timestamp	Number	是	否	时间戳，东八区时间，UTC+8，单位为秒
nonce_str	String	否	是	随机字符串，由开发者设置传入，加强安全性（若不填写可能被重置请求）。随机字符串，不长于32位。推荐使用大小写字母和数字，不同添加请求的nonce_str必须动态生成，若重复将会导致领取失败
fixed_begintimestamp	Number	否	否	卡券在第三方系统的实际领取时间，为东八区时间戳（UTC+8，精确到秒）。当卡券的有效期类为DATE_TYPE_FIX_TERM时专用，标识卡券的实际生效时间，用于解决商户系统内起始时间和领取微信卡券时间不同步的问题
outer_str	String	否	否	领取渠道参数，用于标识本次领取的渠道值
signature	String	是	-	签名，商户将接口列表中的参数按照指定方式进行签名，签名方式使用SHA1

wx.openCard(OBJECT)接口调用成功后，会返回四个值，如下所示。

（1）code：加密 code，为用户领取到卡券的 code 加密后的字符串。

（2）cardId：用户领取到卡券的 Id。

（3）cardExt：用户领取到卡券的扩展参数，与调用时传入的参数相同。

（4）isSuccess：是否成功。

示例代码如下所示。

```
wx.addCard({
  cardList: [
    {
      cardId: '',
      cardExt: '{"code": "", "openid": "", "timestamp": "", "signature":""}'
    }, {
      cardId: '',
      cardExt: '{"code": "", "openid": "", "timestamp": "", "signature":""}'
    }
  ],
  success: function(res) {
```

```
    console.log(res.cardList) // 卡券添加结果
  }
})
```

使用 wx.openCard(OBJECT)添加完卡券后，就可以使用 wx.openCard(OBJECT)来获取卡券列表了，wx.openCard(OBJECT)参数说明如表 3.55 所示。

表 3.55　wx. openCard (OBJECT)参数说明

属　　性	类　　型	必填	说　　明
cardList	ObjectArray	是	需要打开的卡券列表
success	Function	否	接口调用成功的回调函数
fail	Function	否	接口调用失败的回调函数
complete	Function	否	接口调用结束的回调函数（无论调用成功、失败都会执行）

cardList 是一个对象数组，对象里面有两个属性。

（1）cardId：需要打开的卡券 Id。

（2）code：由 addCard 的返回对象中的加密 code 通过解密后得到。

示例代码如下所示。

```
wx.openCard({
  cardList: [
    {
      cardId: '',
      code: ''
    }, {
      cardId: '',
      code: ''
    }
  ],
  success: function(res) {
  }
})
```

第 4 章　微信小程序高级交互设计

微信小程序提供的组件和 API 可以实现高级交互效果，比如海报轮播效果、页签切换效果、水平滑动效果、九宫格导航、列表式导航、区域条件检索设计、底部固定效果设计、图文列表设计、向导型表单设计等，在设计小程序的时候经常会用到这些高级交互效果。

4.1　猫眼电影：海报轮播效果设计

海报轮播效果用到的频率非常高，在有限的区域，可以动态切换展示不同的内容，对于终端小屏来说，确实是不错的选择，所以很多 App 软件和小程序都会采用这样的设计。海报轮播效果需要借助于 swiper 滑块视图容器组件，下面以猫眼电影海报轮播效果为例，如图 4.1 和图 4.2 所示。



图 4.1　海报 1



图 4.2　海报 2

（1）准备好要轮播的图片和放大镜图片。

（2）新建一个项目，名称为猫眼电影，然后进入 app.json 文件，设置导航窗口的颜色以及导航条标题和文字颜色，具体代码如下所示。

```
{
  "pages":[
    "pages/index/index",
    "pages/logs/logs"
  ],
  "window":{
    "backgroundTextStyle": "light",
    "navigationBarBackgroundColor": "#D53E37",
    "navigationBarTitleText": "电影",
    "navigationBarTextStyle": "white"
  }
}
```

（3）进入 pages/index/index.wxml 文件，对搜索框和海报轮播效果界面进行布局，具体代码如下所示。

```
<view class="content">
 <view class="search">
   <view class="searchInput" bindtap="searchInput">
      <image src="/pages/images/fangdajing.jpg" style="width:15px;height:19px;"></image>
      <text class="searchContent">找影视剧、影人、影院</text>
   </view>
 </view>
 <view class="haibao">
   <swiper indicator-dots="{{indicatorDots}}" autoplay="{{autoplay}}" interval="{{interval}}" durati
on="{{duration}}" style="height:74px;">
     <block wx:for="{{imgUrls}}">
        <swiper-item>
          <image src="{{item}}" class="silde-image"></image>
        </swiper-item>
     </block>
   </swiper>
 </view>
 </view>
```

（4）进入 pages/index/index.wxss 样式文件，给搜索框和海报轮播效果界面布局添加样式，具体代码如下所示。

```
.search{
    width: 100%;
```

```
    background-color: #D53E37;
    height: 50px;
    line-height: 50px;
    font-family: "Microsoft YaHei";
}
.searchInput{
    width: 95%;
    background-color: #ffffff;
    margin:0 auto;
    height: 30px;
    line-height: 30px;
    text-align: center;
    border-radius: 15px;
}
.searchInput image{
    position: absolute;
    top:7px;
}
.searchContent{
    font-size:12px;
    color: #777777;
    margin-left: 18px;
}
.haibao{
    text-align: center;
    width: 100%;
}
.silde-image{
    width: 100%;
    height:74px;
}
```

（5）进入 pages/index/index.js 业务逻辑文件，初始化海报轮播效果需要的数据，具体代码如下所示。

```
Page({
  data: {
    indicatorDots: true,
    autoplay: true,
    interval: 5000,
    duration: 1000,
    imgUrls: [
```

```
            "/pages/images/haibao/1.jpg",
            "/pages/images/haibao/2.jpg",
            "/pages/images/haibao/3.jpg",
            "/pages/images/haibao/4.jpg",
            "/pages/images/haibao/5.jpg"
        ]
    }
})
```

这样就完成了猫眼电影海报轮播效果的设计，海报轮播时需要的数据值也可以直接写在 WXML 页面文件里，或者使用 js 业务逻辑处理文件将数据绑定到界面里。

4.2　许鲜全国送：页签切换效果设计

页签切换效果也是经常会用到的一种交互效果，设置多个页签，单击不同的页签，显示对应的内容。这个效果的实现需要使用 swiper 滑块视图容器组件的多个面板，如图 4.3 和图 4.4 所示。



图 4.3　新鲜果蔬



图 4.4　零食饮品

（1）准备好新鲜果蔬和零食饮品的商品图片，新建一个许鲜全国送的项目。

（2）在 app.json 里设置导航条背景色为黄色（# FFD140），文字为全国送，文字颜色为黑色，具体代码如下所示。

```
{
  "pages":[
    "pages/index/index",
    "pages/logs/logs"
  ],
  "window":{
    "backgroundTextStyle":"light",
    "navigationBarBackgroundColor": "#FFD140",
    "navigationBarTitleText": "全国送",
    "navigationBarTextStyle":"black"
  }
}
```

（3）在 pages/index/index.wxml 页面文件里，进行页签切换效果的布局设计，具体代码如下所示。

```
<view class="content">
  <view class="type">
    <view class="{{currentTab==0?'select':'default'}}" data-current="0" bindtap="switchNav">新鲜果蔬</view>
    <view class="{{currentTab==1?'select':'default'}}" data-current="1" bindtap="switchNav">零食饮品</view>
  </view>

  <swiper current="{{currentTab}}" style="height:1500px;">
    <swiper-item>
       <view class="items">
         <view class="item">
           <view><image src="/pages/images/hjs.jpg" style="width:100%"></image></view>
           <view class="title">粉糯甜 de 黄金薯，蒸煮烤随你意！特促价 8 斤仅售 38 元包邮！</view>
           <view class="desc">下单后 2-3 天发货，收货后 24 小时内售后！</view>
           <view class="line"></view>
           <view class="price">￥38.00/份</view>
           <view class="old">￥68.00</view>
         </view>
         <view class="hr"></view>
          <view class="item">
```

```
                <view><image src="/pages/images/txm.jpg" style="width:100%"></image></view>
                <view class="title">抢疯！甜心芒 5 斤 32 块 8，生熟两吃，给你果冻般的香甜 Q 弹！
</view>
                <view class="desc">下单后 2-3 天发货，收货后 24 小时内售后！</view>
                <view class="line"></view>
                <view class="price">￥32.80/份</view>
                <view class="old">￥45.00</view>
            </view>
            <view class="hr"></view>
        </view>
    </swiper-item>
    <swiper-item>
        <view class="items">
            <view class="item">
                <view><image src="/pages/images/mgz.jpg" style="width:100%"></image></view>
                <view class="title">100%果汁，营养好滋味，让你的春天春溢满汁！</view>
                <view class="desc">6 瓶芒果汁+6 瓶红葡萄柚汁，许鲜与福莱士再度合作出品，果味
香浓顺滑！</view>
                <view class="line"></view>
                <view class="price">￥59.00/份</view>
                <view class="old">￥90.00</view>
            </view>
            <view class="hr"></view>
             <view class="item">
                <view><image src="/pages/images/hz.jpg" style="width:100%"></image></view>
                <view class="title">香酥烤灰枣/枣干（约 250g），要的就是这种嘎嘣脆！</view>
                <view class="desc">烤灰枣，香、甜、酥、脆，回味无穷！</view>
                <view class="line"></view>
                <view class="price">￥16.50/份</view>
                <view class="old">￥19.90</view>
            </view>
            <view class="hr"></view>
        </view>
    </swiper-item>
  </swiper>
</view>
```

（4）在 pages/index/index.wxss 样式文件里，给页签切换效果添加样式，具体代码如下所示。

```
.content{
    font-family: "Microsoft YaHei";
```

```
}
.type{
      display: flex;
      flex-direction: row;
      width: 96%;
      margin: 0 auto;
}
.type view{
      margin: 0 auto;
}
.select{
      font-size:14px;
      font-weight: bold;
      width: 25%;
      text-align: center;
      height: 45px;
      line-height: 45px;
      border-bottom:5rpx solid #000000;
}
.default{
      width: 25%;
      font-size:12px;
      text-align: center;
      height: 45px;
      line-height: 45px;
}
.items{
   margin-top:10px;
   background: #DDDDDD;
}
.item{
   background: #ffffff;
   margin-left: 15px;
   margin-right: 15px;
}
.title{
   margin-left: 15px;
   margin-right: 15px;
   font-size: 16px;
   font-weight: bold;
```

```
}
.desc{
   margin-left: 15px;
   margin-right: 15px;
   font-size: 13px;
   color: #666666;
   margin-top:10px;
   margin-bottom: 10px;
}
.line{
   border: 1px solid #cccccc;
   opacity: 0.2;
}
.price{
   margin-left: 15px;
   margin-right: 15px;
   margin-top:10px;
   margin-bottom: 5px;
   font-size: 18px;
   font-weight: bold;
   color: red;
}
.old{
   margin-left: 15px;
   margin-right: 15px;
   margin-top:10px;
   font-size: 13px;
   color: #666666;
   text-decoration: line-through;
}
.hr{
   background: #DDDDDD;
   height: 10px;
}
```

（5）在 pages/index/index.js 业务逻辑处理文件里，初始化数据和添加页签切换事件，具体代码如下所示。

```
Page({
   data: {
      currentTab: 0
   },
```

```
  switchNav: function (e) {
    var page = this;
    if (this.data.currentTab == e.target.dataset.current) {
      return false;
    } else {
      page.setData({ currentTab: e.target.dataset.current });
    }
  }
})
```

4.3　新闻频道：水平滑动效果设计

腾讯新闻的新闻频道水平滑动效果是非常典型的一种交互效果，它通过左右滑动将所有的新闻频道显示出来，这个效果实现需要使用 scroll-view 可滚动视图容器组件，如图 4.5 所示。



图 4.5　腾讯新闻的新闻频道

（1）准备好放大镜图标并新建一个腾讯新闻项目。

（2）在 pages/index/index.wxml 页面文件里，使用 view 组件、scroll-view 组件、image 组件等进行界面布局，具体代码如下所示。

```
<view class="content">
   <view class="navbg">
   <view class="classname"><image src="/pages/images/fangdajing.jpg" class="search"></ima
ge></view>
   <view class="nav">
       <scroll-view class="scroll-view_H" scroll-x="true" >
       <view class="scroll-view_H">
          <view><view class="{{flag==0?'select':'normal'}}" id="0" bindtap="switchNav">要闻</
view></view>
          <view><view class="{{flag==1?'select':'normal'}}" id="1" bindtap="switchNav">科技</
view></view>
          <view><view class="{{flag==2?'select':'normal'}}" id="2" bindtap="switchNav">北京</
view></view>
          <view><view class="{{flag==3?'select':'normal'}}" id="3" bindtap="switchNav">社会</
view></view>
          <view><view class="{{flag==4?'select':'normal'}}" id="4" bindtap="switchNav">财经</
view></view>
          <view><view class="{{flag==5?'select':'normal'}}" id="5" bindtap="switchNav">历史</
view></view>
          <view><view class="{{flag==6?'select':'normal'}}" id="6" bindtap="switchNav">汽车</
view></view>
          <view><view class="{{flag==6?'select':'normal'}}" id="7" bindtap="switchNav">军事</
view></view>
          <view><view class="{{flag==6?'select':'normal'}}" id="8" bindtap="switchNav">游戏</
view></view>
          <view><view class="{{flag==6?'select':'normal'}}" id="9" bindtap="switchNav">体育</
view></view>
       </view>
  </scroll-view>
   </view>
   <view class="add">+</view>
</view>
<swiper current="{{currentTab}}"style="height:800px">
  <swiper-item>
     我是要闻内容
  </swiper-item>
  <swiper-item>
      我是科技内容
```

```
    </swiper-item>
    <swiper-item>
        我是北京内容
    </swiper-item>
    <swiper-item>
        我是社会内容
    </swiper-item>
    <swiper-item>
        我是财经内容
    </swiper-item>
    <swiper-item>
        我是历史内容
    </swiper-item>
    <swiper-item>
        我是汽车内容
    </swiper-item>
    <swiper-item>
        我是军事内容
    </swiper-item>
    <swiper-item>
        我是游戏内容
    </swiper-item>
    <swiper-item>
        我是体育内容
    </swiper-item>
</swiper>
</view>
```

（3）在 pages/index/ndex.wxss 样式文件里，给新闻频道导航添加样式，具体代码如下所示。

```
.search{
    width:  18px;
    height:17px;
    margin-left:10px;
}
.navbg{
    color:  #000000;
    display:  flex;
    flex-direction:  row;
    align-items:  center;
}
```

```
.nav{
   width: 75%;
   height: 36px;
}
.add{
   width: 25%;
   height: 30px;
   line-height: 30px;
   text-align: center;
   margin-right:10px;
   font-size: 40px;
   color: #6BA0D3;
}
.scroll-view_H{
   height: 40px;
   display: flex;
   flex-direction: row;
   margin-left: 5px;
}
.normal{
   width: 30px;
   height: 30px;
   line-height: 30px;
   padding-left:5px;
   padding-right: 5px;
   font-size: 14px;
}
.select{
     width: 30px;
   height: 30px;
   line-height: 30px;
   padding-left:5px;
   padding-right: 5px;
   font-size: 14px;
   font-weight: bold;
   border-bottom:5rpx solid #6BA0D3;
}
```

（4）在 pages/index/index.js 业务逻辑处理文件里，给新闻频道初始化数据并添加新闻频道切换事件，具体代码如下所示。

```
Page({
```

```
  data: {
    currentTab: 0,
    flag: 0
  },
  switchNav: function (e) {
    console.log(e);
    var page = this;
    var id = e.target.id;
    if (this.data.currentTab == id) {
      return false;
    } else {
      page.setData({ currentTab: id });
    }
    page.setData({ flag: id });
  }
})
```

这样就实现了腾讯新闻的新闻频道滑动效果，如图 4.6 和图 4.7 所示。

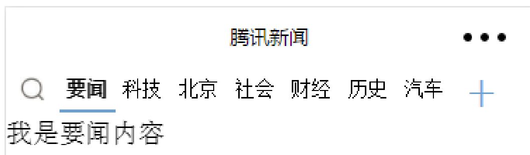


图 4.6　要闻频道

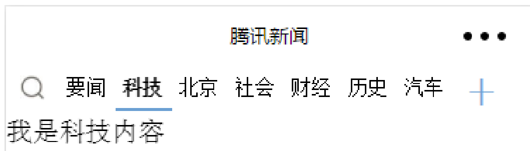


图 4.7　科技频道

4.4　京东到家：手风琴导航切换效果设计

京东到家手风琴导航切换效果是非常典型的一种切换效果，应用非常广泛，比如淘宝、爱鲜蜂等都会用到这样的设计，通过这样的设计，可以将不同类别的商品展现出来，如图 4.8 和图 4.9 所示。



图 4.8　水果蔬菜



图 4.9　肉禽蛋奶

（1）新建一个京东到家项目。

（2）在 pages/index/index.wxml 页面文件里，进行手风琴导航布局设计，具体代码如下所示。

```
<view class="bg">
    <view class="search">
        搜索附近的商品和门店
    </view>
</view>
<view class="content">
  <view class="left">
    <view class="{{flag==0?'select':'normal'}}" id="0" bindtap="switchNav">水果蔬菜</view>
    <view class="{{flag==1?'select':'normal'}}" id="1" bindtap="switchNav">肉禽蛋奶</view>
    <view class="{{flag==2?'select':'normal'}}" id="2" bindtap="switchNav">冷热速食</view>
    <view class="{{flag==3?'select':'normal'}}" id="3" bindtap="switchNav">休闲食品</view>
    <view class="{{flag==4?'select':'normal'}}" id="4" bindtap="switchNav">酒水饮料</view>
    <view class="{{flag==5?'select':'normal'}}" id="5" bindtap="switchNav">粮油调味</view>
    <view class="{{flag==6?'select':'normal'}}" id="6" bindtap="switchNav">清洁日化</view>
    <view class="{{flag==7?'select':'normal'}}" id="7" bindtap="switchNav">家居用品</view>
    <view class="{{flag==8?'select':'normal'}}" id="8" bindtap="switchNav">鲜花蛋糕</view>
    <view class="{{flag==9?'select':'normal'}}" id="9" bindtap="switchNav">医药健康</view>
```

```
    </view>
    <view class="right">
        <view class="category">
            <swiper current="{{currentTab}}" style="height:500px;">
                <swiper-item>
                    我是水果蔬菜内容
                </swiper-item>
                <swiper-item>
                    我是肉禽蛋奶内容
                </swiper-item>
                <swiper-item>
                    我是冷热速食内容
                </swiper-item>
                <swiper-item>
                    我是休闲食品内容
                </swiper-item>
                <swiper-item>
                    我是酒水饮料内容
                </swiper-item>
                <swiper-item>
                    我是粮油调味内容
                </swiper-item>
                <swiper-item>
                    我是清洁日化内容
                </swiper-item>
                <swiper-item>
                    我是家居用品内容
                </swiper-item>
                <swiper-item>
                    我是鲜花蛋糕内容
                </swiper-item>
                <swiper-item>
                    我是医药健康内容
                </swiper-item>
            </swiper>
        </view>
    </view>
</view>
```

（3）在 pages/index/index.wxss 样式文件里，给手风琴导航布局添加样式，具体代码如下所示。

```
.content {
   display: flex;
   flex-direction: row;
   font-family: "Microsoft YaHei";
}
.bg{
   background: #F4F4F4;
   height: 50px;
   display:flex;
   justify-content:center;
   align-items:center;
}
.search{
   width: 96%;
   height: 40px;
   line-height: 40px;
   background: #ffffff;
   margin: 0 auto;
   color:#999999;
}
.left {
   width: 25%;
   font-size: 15px;
   height: 600px;
   background: #F4F4F4;
}

.left view {
   text-align: center;
   height: 45px;
   line-height: 45px;
}
.select{
      background-color: #ffffff;
      border-left:5px solid #36AE66;
      font-weight: bold;
      color: #36AE66;
}
.normal{
      background-color: #F4F4F4;
```

```
    border-bottom: 1px solid #f2f2f2;
}
.right{
    width:75%;
    margin: 10px;
}
```

（4）在 pages/index/index.js 业务逻辑处理文件里，初始化手风琴导航数据并添加导航菜单切换事件，具体代码如下所示。

```
Page({
  data: {
    flag: 0,
    currentTab: 0
  },
  switchNav: function (e) {
    var page = this;
    var id = e.target.id;
    if (this.data.currentTab == id) {
      return false;
    } else {
      page.setData({ currentTab: id });
    }
    page.setData({ flag: id });
  }
})
```

4.5　链家：列表式导航设计

列表式导航是很多 App 或者小程序会采用的一种方式，这种导航方式适合导航比较多的时候，用户可以根据导航名称进入相应的界面，如图 4.10 和图 4.11 所示。



图 4.10　列表式导航一



图 4.11　列表式导航二

（1）准备好列表式导航需要的图片，新建一个链家的项目。

（2）在 pages/index/index.wxml 页面文件里，进行列表式导航布局设计，具体代码如下所示。

```
<view class="content">
  <view class="items">
      <view class="account bg">
        <view class="img">
          <image src="/pages/images/head.jpg" style="width:78px;height:78px;"></image>
        </view>
        <view class="login">
            登录/注册
        </view>
    </view>
    <view class="nav">
        <view class="navName">
          <image src="/pages/images/jjj.jpg" style="width:34px;height:34px;"></image>家基金
        </view>
        <view class="navName">
          <image src="/pages/images/wdkq.jpg" style="width:34px;height:34px;"></image>我的卡券
```

```
        </view>
    </view>

    <view class="hr">
    </view>
    <view class="item">
      <view class="img">
        <image src="/pages/images/xf.jpg" style="width:20px;height:20px;"></image>
      </view>
      <view class="name">我关注的新房</view>
      <view class="detail">
        <text>></text>
        </view>
    </view>
    <view class="line"></view>
    <view class="item">
      <view class="img">
        <image src="/pages/images/esf.jpg" style="width:19px;height:19px;"></image>
      </view>
      <view class="name">我关注的二手房/租房</view>
      <view class="detail">
        <text>></text>
        </view>
    </view>
    <view class="line"></view>
    <view class="item">
      <view class="img">
        <image src="/pages/images/xq.jpg" style="width:21px;height:21px;"></image>
      </view>
      <view class="name">我关注的小区</view>
      <view class="detail">
        <text> ></text>
        </view>
    </view>
    <view class="line"></view>
    <view class="item">
      <view class="img">
        <image src="/pages/images/kfjl.jpg" style="width:19px;height:18px;"></image>
      </view>
      <view class="name">我的看房记录</view>
```

```
    <view class="detail">
      <text> ></text>
      </view>
  </view>
  <view class="line"></view>
  <view class="item">
    <view class="img">
      <image src="/pages/images/ss.jpg" style="width:14px;height:17px;"></image>
    </view>
    <view class="name">我订阅的搜索</view>
    <view class="detail">
      <text> ></text>
      </view>
  </view>
  <view class="line"></view>
  <view class="item">
    <view class="img">
      <image src="/pages/images/dp.jpg" style="width:17px;height:15px;"></image>
    </view>
    <view class="name">我的点评</view>
    <view class="detail">
      <text> ></text>
      </view>
  </view>

  <view class="hr">
  </view>
  <view class="item">
    <view class="img">
      <image src="/pages/images/wd.jpg" style="width:19px;height:19px;"></image>
    </view>
    <view class="name">我的问答</view>
    <view class="detail">
      <text>></text>
      </view>
  </view>

  <view class="hr">
  </view>
  <view class="item">
```

```
    <view class="img">
      <image src="/pages/images/wt.jpg" style="width:19px;height:18px;"></image>
    </view>
    <view class="name">我的委托</view>
    <view class="detail">
      <text>></text>
      </view>
  </view>

  <view class="hr"></view>
  <view class="item">
    <view class="img">
      <image src="/pages/images/yqhy.jpg" style="width:19px;height:18px;"></image>
    </view>
    <view class="name">邀请好友</view>
    <view class="detail">
      <text>></text>
      </view>
  </view>
  <view class="line"></view>
  <view class="item">
    <view class="img">
      <image src="/pages/images/jsq.jpg" style="width:19px;height:19px;"></image>
    </view>
    <view class="name">购房计算器</view>
    <view class="detail">
      <text> ></text>
      </view>
  </view>
  <view class="line"></view>
  <view class="item">
    <view class="img">
      <image src="/pages/images/jjr.jpg" style="width:17px;height:16px;"></image>
    </view>
    <view class="name">我的经纪人</view>
    <view class="detail">
      <text> ></text>
      </view>
  </view>
  <view class="line"></view>
```

```
    <view class="item">
      <view class="img">
        <image src="/pages/images/yhfk.jpg" style="width:19px;height:18px;"></image>
      </view>
      <view class="name">用户反馈</view>
      <view class="detail">
        <text> ></text>
        </view>
    </view>
    <view class="line"></view>
    <view class="item">
      <view class="img">
        <image src="/pages/images/kfdh.jpg" style="width:17px;height:16px;"></image>
      </view>
      <view class="name">客服电话</view>
      <view class="detail">
        <text>10109666 ></text>
        </view>
    </view>
    <view class="line"></view>

    <view class="hr">
    </view>
    <view class="item" bindtap="setting">
      <view class="img">
        <image src="/pages/images/sz.jpg" style="width:20px;height:19px;"></image>
      </view>
      <view class="name">设置</view>
      <view class="detail">
        <text>></text>
        </view>
    </view>
    <view class="line"></view>
  </view>
</view>
```

（3）在 pages/index/index.wxss 样式文件里，给列表式导航布局添加样式，具体代码如下所示。

```
.content{
  font-family: "Microsoft YaHei";
}
```

```
.items{
      background-color: #F0EFF4;
      height:1050px;
}
.account{
   text-align: center;
   padding-top:20px;
   height: 180px;
}
.item{
      display:flex;
      flex-direction:row;
      background-color: #ffffff;
      align-items: center;
}
.bg{
   background: #04A766;
}
.login{
   border: 1px solid #ffffff;
   width:160px;
   height:50px;
   margin:0 auto;
   border-radius: 5px;
   color: #ffffff;
   line-height: 50px;
   font-size: 16px;
}
.nav{
   background-color: #ffffff;
   height: 50px;
   display: flex;
   flex-direction: row;
   text-align: center;
   align-items: center;
}
.navName{
   font-size: 13px;
   width: 50%;
}
```

```
.hr{
    width: 100%;
    height: 10px;
}
.img{
    margin-left:10px;
    line-height: 50px;
}
.name{
    padding-top:15px;
    padding-left: 10px;
    padding-bottom:15px;
    font-size:15px;
}
.detail{
    font-size: 15px;
    position: absolute;
    right: 10px;
    height: 50px;
    line-height: 50px;
    color: #888888;
}
.line{
    border: 1px solid #cccccc;
    opacity: 0.2;
}
```

4.6 支付宝：九宫格导航设计

九宫格导航也被称作宫格导航，是很多 App 软件或者小程序都会采用的一种设计方式，比如支付宝宫格导航、口碑宫格导航、大众点评宫格导航等，通过这种宫格导航设计，可以把重要的内容放到宫格导航里，给用户明确的入口，用户可根据自己的需要进入相关页面，这就是宫格导航的好处，如图 4.12 所示。



图 4.12　支付宝九宫格导航

（1）准备好宫格导航需要的图片，新建一个支付宝项目。

（2）在 pages/index/index.wxml 页面文件里，进行宫格导航的布局设计，具体代码如下所示。

```
<view class="bg">
  <view class="region">
    <view class="search">
      <view class="left">
        <image  src="/pages/images/fdj.jpg"  style="width:16px;height:17px;"></image>热门电影:
神偷奶爸 3</view>
      <view class="right">
        <image src="/pages/images/ht.jpg" style="width:11px;height:15px;"></image>
      </view>
    </view>
    <view class="account">
      <image src="/pages/images/yh.jpg" style="width:24px;height:21px;"></image>
    </view>
    <view class="plus">
      <image src="/pages/images/jh.jpg" style="width:19px;height:19px;"></image>
    </view>
  </view>

  <view class="nav" style="margin-top:20px;">
  <view class="item" bindtap="navBtn">
    <view>
      <image src="/pages/images/sys.jpg" style="width:29px;height:29px;"></image>
```

```
        </view>
        <view style="color:#ffffff;">
          扫一扫
        </view>
      </view>
      <view class="item" bindtap="navBtn">
        <view>
          <image src="/pages/images/fq.jpg" style="width:32px;height:32px;"></image>
        </view>
        <view style="color:#ffffff;">
          付钱
        </view>
      </view>
      <view class="item" bindtap="navBtn">
        <view>
          <image src="/pages/images/sq.jpg" style="width:36px;height:24px;"></image>
        </view>
        <view style="color:#ffffff;">
          收钱
        </view>
      </view>
      <view class="item" bindtap="navBtn">
        <view>
          <image src="/pages/images/kb.jpg" style="width:30px;height:30px;"></image>
        </view>
        <view style="color:#ffffff;">
          卡包
        </view>
      </view>
    </view>
</view>


<view class="nav">
  <block wx:for="{{navs}}">
    <view class="item" bindtap="navBtn" id="{{index}}">
      <view>
        <imagesrc="{{item.img}}"
style="width:{{item.width}}px;height:{{item.height}}px;"></image>
      </view>
```

```
            <view>
                {{item.name}}
            </view>
        </view>
    </block>
</view>
<view class="hr"></view>
```

（3）在 pages/index/index.wxss 样式文件里，给进行宫格导航的布局添加样式，具体代码如下所示。

```
.bg{
    background-color: #1B82D2;
    height: 130px;
}
.region{
    display: flex;
    flex: row;
}
.search{
    width: 70%;
    background-color: #1873B9;
    margin-left: 10px;
    border-radius: 2px;
    display: flex;
    flex-direction: row;
    font-size: 13px;
    color: #ffffff;
    align-items: center;
}
.left{
    width: 90%;
    margin-left: 10px;
}
.right{
    width: 10%;
    text-align: center;
}
.account{
    width:15%;
    text-align: center;
}
```

```
.puls{
  width:15%;
  text-align: center;
}
.nav{
  text-align: left;
}
.item{
    margin-top:15px;
    text-align: center;
    font-family: "Microsoft YaHei";
    font-size: 13px;
    width: 24%;
    display: inline-block;
}
.hr{
  height: 1px;
  background-color: #cccccc;
  opacity: 0.2;
  margin-top:10px;
}
```

（4）在 pages/index/index.js 业务逻辑处理文件里，初始化宫格导航的数据，具体代码如下所示。

```
Page({
  data: {
      navs: []
  },
  onLoad: function (options) {
    var page = this;
    var navs = this.loadNavData();
    page.setData({ navs: navs });
  },
  navBtn: function (e) {
    console.log(e);
    var id = e.currentTarget.id;
    if (id == "9") {
      wx.navigateTo({
        url: '../type/type'
      })
    }
```

```
},
loadNavData: function () {
  var navs = [];
  var nav0 = new Object();
  nav0.img = '/pages/images/hb.jpg';
  nav0.name = '红包';
  nav0.width='19';
  nav0.height='22';
  navs[0] = nav0;

  var nav1 = new Object();
  nav1.img = '/pages/images/zz.jpg';
  nav1.name = '转账';
  nav1.width = '22';
  nav1.height = '23';
  navs[1] = nav1;

  var nav2 = new Object();
  nav2.img = '/pages/images/xyk.jpg';
  nav2.name = '信用卡还款';
  nav2.width = '23';
  nav2.height = '17';
  navs[2] = nav2;

  var nav3 = new Object();
  nav3.img = '/pages/images/czzx.jpg';
  nav3.name = '充值中心';
  nav3.width = '18';
  nav3.height = '23';
  navs[3] = nav3;

  var nav4 = new Object();
  nav4.img = '/pages/images/tp.jpg';
  nav4.name = '淘票票';
  nav4.width = '23';
  nav4.height = '22';
  navs[4] = nav4;

  var nav5 = new Object();
```

```
        nav5.img = '/pages/images/ddcx.jpg';
        nav5.name = '滴滴出行';
        nav5.width = '21';
        nav5.height = '17';
        navs[5] = nav5;

        var nav6 = new Object();
        nav6.img = '/pages/images/yeb.jpg';
        nav6.name = '余额宝';
        nav6.width = '21';
        nav6.height = '23';
        navs[6] = nav6;

        var nav7 = new Object();
        nav7.img = '/pages/images/shh.jpg';
        nav7.name = '生活号';
        nav7.width = '18';
        nav7.height = '21';
        navs[7] = nav7;

        var nav8 = new Object();
        nav8.img = '/pages/images/gxdc.jpg';
        nav8.name = '共享单车';
        nav8.width = '25';
        nav8.height = '21';
        navs[8] = nav8;

        var nav9 = new Object();
        nav9.img = '/pages/images/gd.jpg';
        nav9.name = '更多';
        nav9.width = '17';
        nav9.height = '18';
        navs[9] = nav9;
        return navs;
    }
})
```

4.7　58 同城：区域条件检索设计

在设计微信小程序的时候，经常会涉及区域检索。比如在检索美食的时候，它会让我们先选择区域，然后再选择餐厅。58 同城也有同样的设计，在选择购房的时候，也是有地区、总价、户型、排序四个类别供选择，如图 4.13、图 4.14、图 4.15 和图 4.16 所示。



图 4.13　地区



图 4.14　总价



图 4.15　户型



图 4.16　排序

（1）新建一个 58 同城项目，在 app.json 里设置导航条背景颜色为黑色、导航文字颜色为白色，具体代码如下所示。

```
{
  "pages":[
    "pages/index/index",
    "pages/logs/logs"
  ],
  "window":{
    "backgroundTextStyle":"light",
    "navigationBarBackgroundColor": "#000000",
    "navigationBarTitleText": "58 同城",
    "navigationBarTextStyle":"white"
  }
}
```

（2）在 pages/index/index.wxml 页面文件里，进行区域条件导航菜单的布局设计，具体代码如下所示。

```
<view class="conten">
  <dl class="menu">
     <dt data-index="0" class="{{flag==0?'select':'normal'}}" bindtap="tapMainMenu">地区</dt>
     <dd class="{{subMenuDispaly[0]}}">
```

```
            <ul>
                <li>不限</li>
                <li>朝阳</li>
                <li>海淀</li>
                <li>东城</li>
                <li>西城</li>
                <li>崇文</li>
                <li>宣武</li>
                <li>丰台</li>
                <li>通州</li>
                <li>石景山</li>
                <li>房山</li>
                <li>昌平</li>
                <li>大兴</li>
                <li>顺义</li>
                <li>密云</li>
                <li>怀柔</li>
                <li>延庆</li>
                <li>平谷</li>
                <li>门头沟</li>
            </ul>
        </dd>
        <dt data-index="1" class="{{flag==1?'select':'normal'}}" bindtap="tapMainMenu">总价</dt>
        <dd class="{{subMenuDispaly[1]}}">
            <ul>
                <li>不限</li>
                <li>200 万以下</li>
                <li>200-250 万</li>
                <li>250-300 万</li>
                <li>300-350 万</li>
                <li>350-400 万</li>
                <li>400-500 万</li>
                <li>500-700 万</li>
                <li>700-1000 万</li>
                <li>1000 万以上</li>
            </ul>
        </dd>
        <dt data-index="2" class="{{flag==2?'select':'normal'}}" bindtap="tapMainMenu">户型</dt>
        <dd class="{{subMenuDispaly[2]}}">
            <ul>
```

```
                <li>不限</li>
                <li>一室</li>
                <li>二室</li>
                <li>三室</li>
                <li>四室</li>
                <li>四室以上</li>
            </ul>
        </dd>
        <dt data-index="3" class="{{flag==3?'select':'normal'}}" bindtap="tapMainMenu">排序</dt>
        <dd class="{{subMenuDispaly[3]}}">
            <ul>
                <li>不限</li>
                <li>最近发布</li>
                <li>距离由近到远</li>
                <li>面积由大到小</li>
                <li>总价由低到高</li>
            </ul>
        </dd>
    </dl>
</view>
```

（3）在 pages/index/index.wxss 样式文件里，给区域条件导航菜单的布局添加样式，具体代码如下所示。

```
.content {
    width: 100%;
}

.menu{
    display: block;
    height: 38px;
}
.select{
    font-size: 13px;
    float: left;
    width: 24.7%;
    height: 38px;
    border-right: 1px solid #f2f2f2;
    border-bottom: 2px solid #ff0000;
    text-align: center;
    line-height: 38px;
    background-color: #FAFAFA;
```

```
    color: #666666;

}
.normal{
    font-size: 13px;
    float: left;
    width: 24.7%;
    height: 38px;
    border-right: 1px solid #f2f2f2;
    border-bottom: 1px solid #f2f2f2;
    text-align: center;
    line-height: 38px;
    background-color: #FAFAFA;
    color: #666666;

}
.menu dd{
    position: absolute;
    width: 100%;
    top:40px;
    left: 0;
    z-index: 999;
    background-color: #FAFAFA;
}
.menu li{
    font-size: 14px;
    line-height: 34px;
    color: #666666;
    display: block;
    padding-left: 30px;
    border-bottom:1px solid #f2f2f2;
}
.show{
    display: block;
}
.hidden{
    display: none;
}
```

（4）在 pages/index/index.js 业务逻辑处理文件里，初始化区域条件导航菜单数据并添加菜单切换事件，具体代码如下所示。

```
function initSubMenuDisplay() {
   return ['hidden', 'hidden', 'hidden', 'hidden'];
}
Page({
   data: {
         subMenuDispaly: initSubMenuDisplay(),
         flag:0
   },
   tapMainMenu: function (e) {
      console.log(e);
      var index = parseInt(e.currentTarget.dataset.index);
      console.log(index);
      var newSubMenuDisplay = initSubMenuDisplay();
      if (this.data.subMenuDispaly[index] == 'hidden') {
         newSubMenuDisplay[index] = 'show';
      } else {
         newSubMenuDisplay[index] = 'hidden';
         index = -1;
      }
      this.setData({ subMenuDispaly: newSubMenuDisplay,flag:index });
   }
})
```

4.8 12306 火车票：底部固定效果设计

12306 火车票底部放置了五个导航页签，它们不会随着页面的滚动而滚动，这种效果就是固定在底部的效果，这五个导航页签是：发时、历时、到时、票价、余票和如图 4.17 所示。

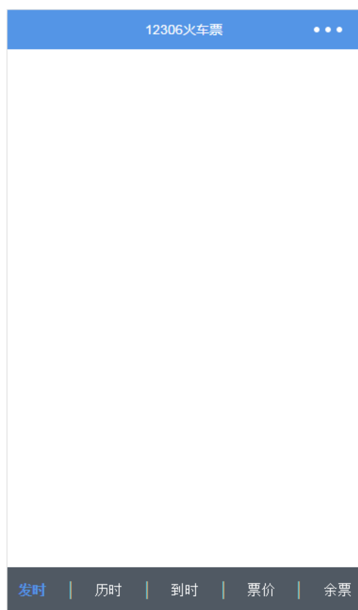


图 4.17　底部固定效果

（1）新建一个 12306 火车票项目，在 app.json 文件设置导航条背景颜色为蓝色（5495E6），导航文字为 12306 火车票，文字颜色为白色，具体代码如下所示。

```
{
  "pages":[
    "pages/index/index",
    "pages/logs/logs"
  ],
  "window":{
    "backgroundTextStyle":"light",
    "navigationBarBackgroundColor": "#5495E6",
    "navigationBarTitleText": "12306 火车票",
    "navigationBarTextStyle":"white"
  }
}
```

（2）在 pages/index/index.wxml 页面文件里，进行底部导航固定效果的布局设计，具体代码如下所示。

```
<view  class="bottomNav">
      <view  id="0"  class="{{currentTab==0?'selected':'common'}}"  bindtap="switchNav">发时</vi
ew>
      <view  style="color:#ffffff">|</view>
```

```
        <view id="1" class="{{currentTab==1?'selected':'common'}}" bindtap="switchNav">历时</view>
        <view style="color:#ffffff">|</view>
        <view id="2" class="{{currentTab==2?'selected':'common'}}" bindtap="switchNav">到时</view>
        <view style="color:#ffffff">|</view>
        <view id="3" class="{{currentTab==3?'selected':'common'}}" bindtap="switchNav">票价</view>
        <view style="color:#ffffff">|</view>
        <view id="4" class="{{currentTab==4?'selected':'common'}}" bindtap="switchNav">余票</view>
    </view>
```

（3）在 pages/index/index.wxss 样式文件里，给底部导航固定效果添加样式，具体代码如下所示。

```
.bottomNav{
    background-color: #505963;
    display: flex;
    flex-direction: row;
    height: 50px;
    line-height: 50px;
    position: fixed;
    bottom:0px;
    width: 100%;
}
.bottomNav view{
    margin: 0 auto;
}
.selected{
    font-size: 15px;
    color: #5495E6;
    font-weight: bold;

}
.common{
    font-size: 15px;
    color: #ffffff;
}
```

（4）在 pages/index/index.js 业务逻辑处理文件里，初始化底部固定效果数据并切换事件绑定，具体代码如下所示。

```
Page({
  data: {
    currentTab: '1'
  },
  switchNav: function (e) {
    var id = e.currentTarget.id;
    console.log(id);
    this.setData({ currentTab: id });
  }
})
```

4.9　当当购物：图文列表设计

图文列表设计是很多 App 和小程序都会采用的一种布局方式，比如猫眼电影采用图文的方式展示电影信息、拉勾网采用图文的方式展示企业招聘信息，由此可见，图文列表展现是非常受欢迎的一种展现方式。下面以当当购物为例，看看图文列表是如何设计的，如图 4.18 所示。



图 4.18　当当购物

（1）准备好图文列表需要的图片，新建一个当当购物的项目，在 app.json 里设置导航条背景颜色为红色（#E53D30），导航文字为当当购物，文字颜色为白色，具体代码如下所示。

```
{
  "pages":[
    "pages/index/index",
    "pages/logs/logs"
  ],
  "window":{
    "backgroundTextStyle":"light",
    "navigationBarBackgroundColor": "#E53D30",
    "navigationBarTitleText": "当当购物",
    "navigationBarTextStyle":"white"
  }
}
```

（2）在 pages/index/index.wxml 页面文件里，进行图文列表的布局设计，具体代码如下所示。

```
<view class="content">
 <view class="search">
   <view class="searchInput" bindtap="searchInput">
      <image src="/pages/images/fdj.jpg" style="width:18px;height:16px;"></image>
      <text class="searchContent">搜索当当商品</text>
   </view>
 </view>
<view class="type">
    <view class="{{currentTab==0?'select':'default'}}" data-current="0" bindtap="switchNav">新书热卖榜</view>
    <view class="{{currentTab==1?'select':'default'}}" data-current="1" bindtap="switchNav">图书畅销榜</view>
    <view class="{{currentTab==2?'select':'default'}}" data-current="2" bindtap="switchNav">童书榜</view>
</view>
<view class="hr"></view>
<view>
  <swiper current="{{currentTab}}" style="height:1000px;">
    <swiper-item>
       <view class="list">
         <view class="book">
              <view class="pic">
                  <image src="/pages/images/num1.jpg" mode="aspectFit" style="width:115p
```

```
x;height:120px;"></image>
                </view>
                <view class="movie-info">
                    <view class="base-info">
                        <view class="name">《红星照耀中国》</view>
                        <view class="desc">作者：埃德加・斯诺 著；</view>
                        <view class="desc">出版社：人民文学出版社</view>
                        <view class="desc">出版时间：2017 年 6 月</view>
                        <view class="people"><text class="price">￥26.10</text><text class="o
rg">￥26.10</text> <text>7.91 折</text></view>
                            <view class="desc">
                            <image src="/pages/images/star.jpg" style="width:12px;height:12px;
"></image>
                            <image src="/pages/images/star.jpg" style="width:12px;height:12px;
"></image>
                            <image src="/pages/images/star.jpg" style="width:12px;height:12px;
"></image>
                            <image src="/pages/images/star.jpg" style="width:12px;height:12px;
"></image>
                            <image src="/pages/images/star.jpg" style="width:12px;height:12px;
"></image>
                            20 条评论</view>
                    </view>
                </view>
            </view>
        <view class="hr"></view>

        <view class="book">
                <view class="pic">
                    <image src="/pages/images/num2.jpg" mode="aspectFit" style="width:115p
x;height:120px;"></image>
                </view>
                <view class="movie-info">
                    <view class="base-info">
                        <view class="name">跃迁：成为高手的技术</view>
                        <view class="desc">作者：古典</view>
                        <view class="desc">出版社：中信出版社</view>
                        <view class="desc">出版时间：2017 年 8 月</view>
                        <view class="people"><text class="price">￥36.10</text><text class="o
rg">￥49.10</text> <text>7.51 折</text></view>
                            <view class="desc">
```

```
                        <image src="/pages/images/star.jpg" style="width:12px;height:12px;
"></image>
                        <image src="/pages/images/star.jpg" style="width:12px;height:12px;
"></image>
                        <image src="/pages/images/star.jpg" style="width:12px;height:12px;
"></image>
                        <image src="/pages/images/star.jpg" style="width:12px;height:12px;
"></image>
                        17 条评论</view>
                    </view>
                </view>
            </view>
        <view class="hr"></view>

        <view class="book">
                <view class="pic">
                    <image src="/pages/images/num3.jpg" mode="aspectFit" style="width:115p
x;height:120px;"></image>
                </view>
                <view class="movie-info">
                    <view class="base-info">
                        <view class="name">心有所定，不畏浮世</view>
                        <view class="desc">作者：张德芬</view>
                        <view class="desc">出版社：北京联合出版有限公司</view>
                        <view class="desc">出版时间：2017 年 7 月</view>
                        <view class="people"><text class="price">￥27.50</text><text class="o
rg">￥39.80</text> <text>6.91 折</text></view>
                        <view class="desc">
                        <image src="/pages/images/star.jpg" style="width:12px;height:12px;
"></image>
                        <image src="/pages/images/star.jpg" style="width:12px;height:12px;
"></image>
                        <image src="/pages/images/star.jpg" style="width:12px;height:12px;
"></image>
                        <image src="/pages/images/star.jpg" style="width:12px;height:12px;
"></image>
                        15 条评论</view>
                    </view>
                </view>
            </view>
```

```
        <view class="hr"></view>

        <view class="book">
                <view class="pic">
                    <image src="/pages/images/num4.jpg" mode="aspectFit" style="width:115p
x;height:120px;"></image>
                </view>
                <view class="movie-info">
                    <view class="base-info">
                        <view class="name">红星照耀中国（团购更优惠）</view>
                        <view class="desc">作者：[美]埃德加・斯诺</view>
                        <view class="desc">出版社：人民文学出版社</view>
                        <view class="desc">出版时间：2017 年 6 月</view>
                        <view class="people"><text class="price">￥34.00</text><text class="o
rg">￥43.10</text> <text>7.91 折</text></view>
                          <view class="desc">
                          <image src="/pages/images/star.jpg" style="width:12px;height:12px;
"></image>
                          <image src="/pages/images/star.jpg" style="width:12px;height:12px;
"></image>
                          <image src="/pages/images/star.jpg" style="width:12px;height:12px;
"></image>
                          <image src="/pages/images/star.jpg" style="width:12px;height:12px;
"></image>
                          <image src="/pages/images/star.jpg" style="width:12px;height:12px;
"></image>
                          163 条评论</view>
                    </view>
                </view>
            </view>
        <view class="hr"></view>

        <view class="book">
                <view class="pic">
                    <image src="/pages/images/num5.jpg" mode="aspectFit" style="width:115p
x;height:120px;"></image>
                </view>
                <view class="movie-info">
                    <view class="base-info">
                        <view class="name">外婆的道歉信</view>
```

```
                        <view class="desc">作者：[瑞典]弗雷德里克・巴克曼</view>
                        <view class="desc">出版社：天津人民出版社</view>
                        <view class="desc">出版时间：2017 年 5 月</view>
                        <view class="people"><text class="price">￥31.20</text><text class="o
rg">￥42.10</text> <text>7.43 折</text></view>
                          <view class="desc">
                          <image src="/pages/images/star.jpg" style="width:12px;height:12px;
"></image>
                          <image src="/pages/images/star.jpg" style="width:12px;height:12px;
"></image>
                          <image src="/pages/images/star.jpg" style="width:12px;height:12px;
"></image>
                          <image src="/pages/images/star.jpg" style="width:12px;height:12px;
"></image>
                          <image src="/pages/images/star.jpg" style="width:12px;height:12px;
"></image>
                          2391 条评论</view>
                    </view>
                </view>
           </view>
        <view class="hr"></view>

        <view class="book">
                <view class="pic">
                    <image src="/pages/images/num6.jpg" mode="aspectFit" style="width:115p
x;height:120px;"></image>
                </view>
                <view class="movie-info">
                    <view class="base-info">
                        <view class="name">你不必向这个世界证明什么</view>
                        <view class="desc">作者：吕峥</view>
                        <view class="desc">出版社：台湾出版社</view>
                        <view class="desc">出版时间：2017 年 7 月</view>
                        <view class="people"><text class="price">￥23.30</text><text class="o
rg">￥39.80</text> <text>5.85 折</text></view>
                          <view class="desc">
                          <image src="/pages/images/star.jpg" style="width:12px;height:12px;
"></image>
                          <image src="/pages/images/star.jpg" style="width:12px;height:12px;
"></image>
```

```
                        <image src="/pages/images/star.jpg" style="width:12px;height:12px;
"></image>
                        <image src="/pages/images/star.jpg" style="width:12px;height:12px;
"></image>
                        <image src="/pages/images/star.jpg" style="width:12px;height:12px;
"></image>
                        1230 条评论</view>
                  </view>
              </view>
         </view>
      <view class="hr"></view>


 </view>
    </swiper-item>
    <swiper-item>
    </swiper-item>
    <swiper-item>
    </swiper-item>
  </swiper>
</view>
</view>
```

（3）在 pages/index/index.wxss 样式文件里，给图文列表的布局添加样式，具体代码如下所示。

```
.content{
   font-family: "Microsoft YaHei";
}
.search{
    width: 100%;
    background-color: #E53D30;
    height: 50px;
    line-height: 50px;
    font-family: "Microsoft YaHei";
    position: fixed;
    z-index: 999;
}
.searchInput{
    width: 95%;
    background-color: #ffffff;
    margin:0 auto;
```

```
    height: 30px;
    line-height: 30px;
    border-radius: 15px;
}
.searchInput image{
    position: absolute;
    top:7px;
    left:15px;
}
.searchContent{
    font-size:12px;
    color: #777777;
    margin-left: 30px;
}
.type{
    display: flex;
    flex-direction: row;
    width: 100%;
    margin: 0 auto;
    position: fixed;
    z-index: 999;
    background: #f2f2f2;
    top:50px;
}
.type view{
    margin: 0 auto;
}
.select{
    font-size:16px;
    font-weight: bold;
    width: 25%;
    text-align: center;
    height: 45px;
    line-height: 45px;
    border-bottom:5rpx solid #E53D30;
    color: #E53D30;
}
.default{
    width: 25%;
    font-size:16px;
```

```
        text-align: center;
        height: 45px;
        line-height: 45px;
}
.list{
    margin-top: 100px;
}
.book{
        display: flex;
        flex-direction: row;
        width: 100%;
}
.pic image{
        width:80px;
        height:100px;
        padding:10px;
}
.base-info{
        font-size: 12px;
        padding-top: 10px;
        line-height: 20px;
}
.name{
        font-size: 13px;
        font-weight: bold;
        color: #000000;
}
.people{
        color: #555555;
        margin-top: 5px;
        margin-bottom: 5px;
}
.price{
        font-size: 18px;
        font-weight: bold;
        color: #E53D30;
        margin-left:5px;
}
.org{
    text-decoration: line-through;
```

```
  margin-left: 5px;
  margin-right: 5px;
}
.desc{
    color: #333333;
}
.hr{
    height: 1px;
    width: 100%;
    background-color: #cccccc;
    opacity: 0.2;
}
.btn{
    position: absolute;
    right: 10px;
    margin-top:50px;
}
.btn button{
    width:52px;
    height: 25px;
    font-size:11px;
    color: red;
    border: 1px solid red;
    background-color: #ffffff;
}
```

（4）在 pages/index/index.js 业务逻辑处理文件里，初始化页签导航需要的数据以及页签切换的绑定事件，具体代码如下所示。

```
Page({
  data: {
    currentTab: 0
  },
  switchNav: function (e) {
    var page = this;
    if (this.data.currentTab == e.target.dataset.current) {
      return false;
    } else {
      page.setData({ currentTab: e.target.dataset.current });
    }
  }
})
```

4.10　中信信用卡申请：向导型表单设计

微信小程序里经常会涉及表单，填写表单数据，然后将数据保存到本地或者传输到后台服务器里保存起来，当待填写的表单数据项较多的时候，一般会采用向导型表单设计的方式来分步骤填写表单项，如图 4.19、图 4.20、图 4.21 和图 4.22 所示。

图 4.19　详细信息 1

图 4.20　详细信息 2



图 4.21　联系人信息



图 4.22　表单校验

1．基本信息设计

（1）新建一个中信信用卡项目，在 app.json 里设置导航条背景色为浅灰色（#f2f2f2），导航文字为中信银行信用卡，具体代码如下所示。

```
{
  "pages":[
    "pages/index/index",
    "pages/logs/logs"
  ],
  "window":{
    "backgroundTextStyle":"light",
    "navigationBarBackgroundColor": "#f2f2f2",
    "navigationBarTitleText": "中信银行信用卡",
    "navigationBarTextStyle":"black"
  }
}
```

（2）在 pages/index/index.wxml 页面文件里设计基本信息，使用 view 视图容器组件、表单组件等进行界面布局设计，具体代码如下所示。

```
<view class="content">
<view class="tip {{info}}">{{tip}}</view>
```

```
<view class="items">
    <view class="item">
        <view><image src="/pages/images/jbxx-1.jpg" style="width:35px;height:35px;"></image>
</view>
        <view class="select">基本信息</view>
    </view>
    <view class="item dh">
        <view><image src="/pages/images/dh-1.jpg" style="width:14px;height:16px;"></image></
view>
    </view>
    <view class="item">
        <view><image src="/pages/images/xxxx-0.jpg" style="width:35px;height:35px;"></image>
</view>
        <view class="default">详细信息</view>
    </view>
    <view class="item dh">
        <view><image src="/pages/images/dh-0.jpg" style="width:14px;height:16px;"></image></
view>
    </view>
    <view class="item">
        <view><image src="/pages/images/lxrxx-0.jpg" style="width:35px;height:35px;"></image>
</view>
        <view class="default">联系人信息</view>
    </view>
</view>
<view class="menu">
身份证信息
</view>
<view>
   <form bindsubmit="formSubmit">
   <view class="category">
      <view class="title">姓名</view>
      <view class="input"><input name="name" placeholder="不少于 2 个汉字" placeholder-style
="color:#999999;font-size:16px;" bindblur="checkValue"/></view>
   </view>
   <view class="line"></view>

   <view class="category">
      <view class="title">姓名拼音</view>
      <view class="input"><input name="pinyin"  placeholder="请输入姓名拼音" placeholder-styl
```

```
e="color:#999999;font-size:16px;"  bindblur="checkValue"/></view>
  </view>
  <view  class="line"></view>

   <view  class="category">
     <view  class="title">身份证号</view>
     <view  class="input"><input  name="idcard"  placeholder="请输入身份证号"  type="idcard"  pl
aceholder-style="color:#999999;font-size:16px;"  bindblur="checkValue"/></view>
  </view>
  <view  class="line"></view>

  <view  class="category">
    <view  class="title">发证机关</view>
    <view  class="input"><input  name="dept"  placeholder="发证机关位于身份证背面"   placehol
der-style="color:#999999;font-size:16px;"  bindblur="checkValue"/></view>
  </view>
  <view  class="line"></view>

  <view  class="category">
    <view  class="title">有效期  自</view>
    <view  class="input">
        <picker  mode="date"  value="{{dateStart}}"  name="dateStart"  start="2015-09-01"  end="
2017-09-01"  bindchange="bindStartDateChange">
    <view  class="picker">
      {{dateStart}}
    </view>
  </picker>
    </view>
  </view>
  <view  class="line"></view>

  <view  class="category">
    <view  class="title">有效期  至</view>
    <view  class="input">
        <picker  mode="date"  value="{{dateStart}}"  name="dateEnd"  start="2015-09-01"  end="2
017-09-01"  bindchange="bindEndDateChange">
    <view  class="picker">
      {{dateEnd}}
    </view>
  </picker>
```

```
        </view>
    </view>
    <view class="line"></view>
<view class="hr"></view>
    <view class="category">
        <view class="title">邮箱</view>
        <view class="input"><input name="email" placeholder="电子账单将寄往该邮箱地址" place
holder-style="color:#999999;font-size:16px;" bindblur="checkValue"/></view>
    </view>
    <view class="line"></view>

      <view class="category">
        <view class="title">手机号码</view>
        <view class="input"><input  name="phone" placeholder="11 位手机号码" maxlength="11"
    type="数字输入键盘" placeholder-style="color:#999999;font-size:16px;" bindblur="checkValue"
/></view>
    </view>
    <view class="line"></view>
    <button class="btn" form-type="submit">下一步</button>
    </form>
</view>

</view>
```

（3）在 pages/index/index.wxss 样式文件里，给基本信息布局添加样式，具体代码如下所示。

```
.content{
      font-family: "Microsoft YaHei";
}
.items{
    display: flex;
    flex-direction: row;
    margin: 0 auto;
    text-align: center;
    align-items: center;
    margin-top:20px;
    margin-bottom: 20px;
}
.item{
    width: 30%;
    font-size: 16px;
```

```
}
.dh{
   width:  5%;
}
.default{
   color:  #999999;
}
.menu{
   height:  50px;
   background-color:  #f2f2f2;
   line-height:  50px;
   padding-left:  15px;
}
.category{
   display:  flex;
   flex-direction:  row;
   margin-top:10px;
   margin-bottom:  10px;
   margin-left:  15px;
}
.title{
   width:30%;
}
.input{
   width:70%;
}
.line{
   border:  1px  solid  #cccccc;
   opacity:  0.2;
}
.btn{
   background-color:  #E64340;
   color:  #ffffff;
   margin-top:5px;
   width:90%;
}
.hr{
   height:  10px;
   background-color:  #f2f2f2;
}
```

```
.tip{
  height: 40px;
  background-color: #E64340;
  color: #ffffff;
  font-size: 16px;
  text-align: center;
  line-height: 40px;
}
.hide{
  display: none;
}
```

（4）在 pages/index/index.js 业务逻辑处理文件里进行表单校验，并将数据缓存到本地，在 app.json 里新增一个详细信息页面“pages/detail/detail”，单击“下一步”按钮的时候就会跳转到详细信息页面，具体代码如下所示。

```
Page({
  data:{
    dateStart:'YYYY-MM-DD',
    dateEnd: 'YYYY-MM-DD',
    info:'hide',
    tip:''
  },
  bindStartDateChange: function (e) {
    this.setData({
      dateStart: e.detail.value
    })
  },
  bindEndDateChange: function (e) {
    this.setData({
      dateEnd: e.detail.value
    })
  },
  formSubmit:function(e){
    console.log(e);
     var page = this;
     var card = new Object();
     var name = e.detail.value.name;
     if (name == "") {
       page.setData({ info: 'show', tip: '姓名请输入中文，不少于 2 个汉字' })
       return;
     } else {
```

```
    card.name = name;
  }

  var pinyin = e.detail.value.pinyin;
  if (pinyin == "") {
    page.setData({ info: 'show', tip:'姓名拼音不能为空且不得少于 4 个字符' })
    return;
  } else {
    card.pinyin = pinyin;
  }


  var idcard = e.detail.value.idcard;
  if (idcard == "") {
    page.setData({ info: 'show', tip: '请输入身份证号码！' })
    return;
  } else {
    card.idcard = idcard;
  }

  var dept = e.detail.value.dept;
  if (dept == "") {
    page.setData({ info: 'show', tip: '请输入发证机关！' })
    return;
  } else {
    card.dept = dept;
  }

  var dateStart = e.detail.value.dateStart;

  var dateEnd = e.detail.value.dateEnd;

  var email = e.detail.value.email;
  if (email == "") {
    page.setData({ info: 'show', tip: '请输入正确的邮箱！' })
    return;
  } else {
    card.email = email;
  }
```

```
        var phone = e.detail.value.phone;
        if (phone == "") {
            page.setData({ info: 'show', tip: '请输入正确的 11 位手机号码！！' })
            return;
        } else {
            card.phone = phone;
        }
        wx.setStorageSync("card", card);
        wx.redirectTo({
            url: '../detail/detail',
        })
    },
    checkValue:function(e){
        console.log(e);
        var name = e.detail.value;
        if(name !=''){
            this.setData({info:'hide'});
        }
    }
})
```

界面效果和表单数据如图 4.23 和图 4.24 所示。

图 4.23　基本信息设计

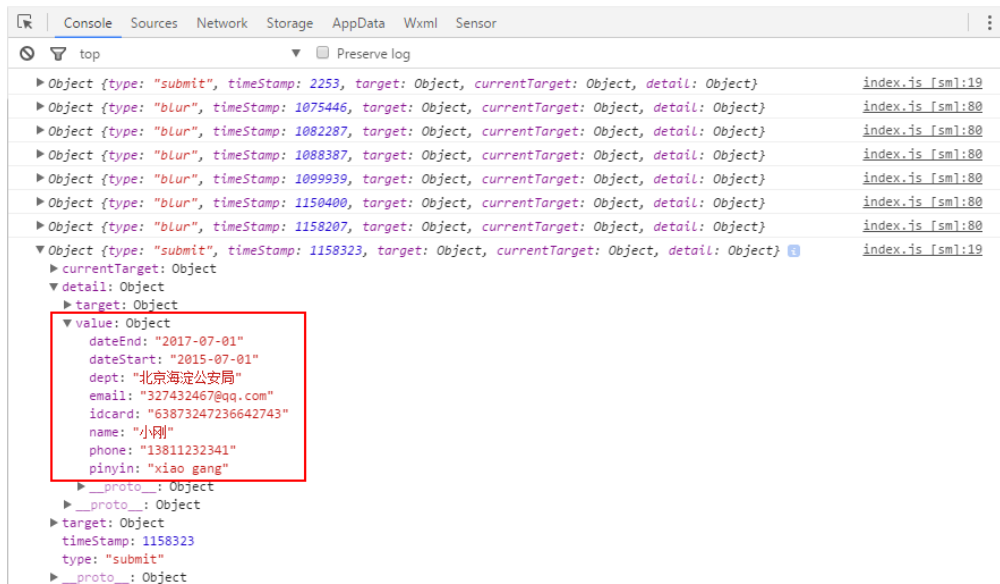


图 4.24　基本信息数据

2．详细信息设计

（1）在 pages/detail/detail.wxml 页面文件里设计详细信息，使用 view 视图容器组件、表单组件等进行界面布局设计，具体代码如下所示。

```
<view class="content">
<view class="tip {{info}}">{{tip}}</view>
<view class="items">
    <view class="item">
        <view><image src="/pages/images/jbxx-1.jpg" style="width:35px;height:35px;"></image></view>
        <view class="select">基本信息</view>
    </view>
    <view class="item dh">
        <view><image src="/pages/images/dh-1.jpg" style="width:14px;height:16px;"></image></view>
    </view>
    <view class="item">
        <view><image src="/pages/images/xxxx-1.jpg" style="width:35px;height:35px;"></image></view>
        <view class="select">详细信息</view>
    </view>
```

```
    <view class="item dh">
        <view><image src="/pages/images/dh-0.jpg" style="width:14px;height:16px;"></image></
view>
    </view>
    <view class="item">
        <view><image src="/pages/images/lxrxx-0.jpg" style="width:35px;height:35px;"></image>
</view>
        <view class="default">联系人信息</view>
    </view>
</view>
<view class="menu">
工作单位
</view>
<view>
  <form bindsubmit="formSubmit">
  <view class="category">
    <view class="title">工作单位</view>
    <view class="input"><input name="deptName" placeholder="不少于 4 个汉字" placeholder-
style="color:#999999;font-size:16px;" bindblur="checkValue"/></view>
  </view>
  <view class="line"></view>

  <view class="category">
    <view class="title">单位地址</view>
    <view class="input"><input name="deptAddress"  placeholder="请输入单位地址" placehold
er-style="color:#999999;font-size:16px;" bindblur="checkValue"/></view>
  </view>
  <view class="line"></view>

    <view class="category">
    <view class="title">职位/岗位</view>
    <view class="input"><input name="position" placeholder="职位/岗位" type="idcard" placeh
older-style="color:#999999;font-size:16px;" bindblur="checkValue"/></view>
  </view>
  <view class="line"></view>
<view class="menu">
工作单位
</view>
  <view class="category">
    <view class="title">婚姻状况</view>
```

```
      <view class="input">
      <radio-group class="radio-group"  bindchange="radioChange">
            <radio name="marry" value="已婚" checked/>已婚
            <radio name="marry" value="未婚" />未婚
</radio-group>
      </view>
   </view>
   <view class="line"></view>

   <view class="category">
      <view class="title">学历</view>
      <view class="input">
          <view bindtap="selectEdu">{{edu}}</view>
      </view>
   </view>
   <view class="line"></view>
   <view class="category">
      <view class="title">居住地址</view>
      <view class="input"><input name="address" placeholder="现居住的详细地址"  placeholder
-style="color:#999999;font-size:16px;" bindblur="checkValue"/></view>
   </view>
   <view class="line"></view>
   <view class="hr"></view>
   <button class="btn" form-type="submit">下一步</button>
   </form>
</view>

</view>
```

（2）在 pages/detail/detail.wxss 样式文件里，给详细信息布局添加样式，具体代码如下所示。

```
.content{
    font-family: "Microsoft YaHei";
}
.items{
   display: flex;
   flex-direction: row;
   margin: 0 auto;
   text-align: center;
   align-items: center;
   margin-top:20px;
```

```
    margin-bottom: 20px;
}
.item{
    width: 30%;
    font-size: 16px;
}
.dh{
    width: 5%;
}
.default{
    color: #999999;
}
.menu{
    height: 50px;
    background-color: #f2f2f2;
    line-height: 50px;
    padding-left: 15px;
}
.category{
    display: flex;
    flex-direction: row;
    margin-top:10px;
    margin-bottom: 10px;
    margin-left: 15px;
}
.title{
    width:30%;
}
.input{
    width:70%;
}
.line{
    border: 1px solid #cccccc;
    opacity: 0.2;
}
.btn{
    background-color: #E64340;
    color: #ffffff;
    margin-top:5px;
    width:90%;
```

```
}
.hr{
   height: 10px;
   background-color: #f2f2f2;
}
.tip{
   height: 40px;
   background-color: #E64340;
   color: #ffffff;
   font-size: 16px;
   text-align: center;
   line-height: 40px;
}
.hide{
   display: none;
}
```

（3）在 pages/detail/detail.js 业务逻辑处理文件里进行表单校验，并将数据缓存到本地，在 app.json 里新增一个联系人信息页面“pages/contact/contact”，单击“下一步”按钮的时候会跳转到联系人信息页面，具体代码如下所示。

```
Page({
   data: {
      info: 'hide',
      tip: '',
      edu:'请选择您的最高学历',
      marry:''
   },
   formSubmit: function (e) {
      console.log(e);
      var page = this;
      var card = wx.getStorageSync("card");
      var deptName = e.detail.value.deptName;
      if (deptName == "") {
         page.setData({ info: 'show', tip: '单位名称不少于 4 个汉字或 8 个字符' })
         return;
      } else {
         card.deptName = deptName;
      }

      var deptAddress = e.detail.value.deptAddress;
      if (deptAddress == "") {
```

```
      page.setData({ info: 'show', tip: '请输入单位地址！' })
      return;
    } else {
      card.deptAddress = deptAddress;
    }


    var position = e.detail.value.position;
    if (position == "") {
      page.setData({ info: 'show', tip: '请输入职位或者岗位名称！' })
      return;
    } else {
      card.position = position;
    }

    var address = e.detail.value.address;
    if (address == "") {
      page.setData({ info: 'show', tip: '请输入现居住的详细地址！' })
      return;
    } else {
      card.address = address;
    }

    card.edu= this.data.edu;
    card.marry = this.data.marry;
    wx.setStorageSync("card", card);
    wx.redirectTo({
      url: '../contact/contact',
    })
  },
  checkValue: function (e) {
    var name = e.detail.value;
    if (name != '') {
      this.setData({ info: 'hide' });
    }
  },
  radioChange:function(e){
      var page = this;
      page.setData({marry:e.detail.value});
  },
```

```
  selectEdu:function(){
    var page = this;
    var edu = ['硕士及以上', '本科', '大专', '高中或中专','初中或以下'];
    wx.showActionSheet({
      itemList: edu,
      success: function (res) {
        if (!res.cancel) {
          console.log(res.tapIndex);//数组序号
          console.log(edu[res.tapIndex]);//数组序号对应的值
          page.setData({ edu: edu[res.tapIndex] });
        }
      }
    })
  }
})
```

界面效果和表单数据如图 4.25 和图 4.26 所示。

图 4.25　详细信息设计

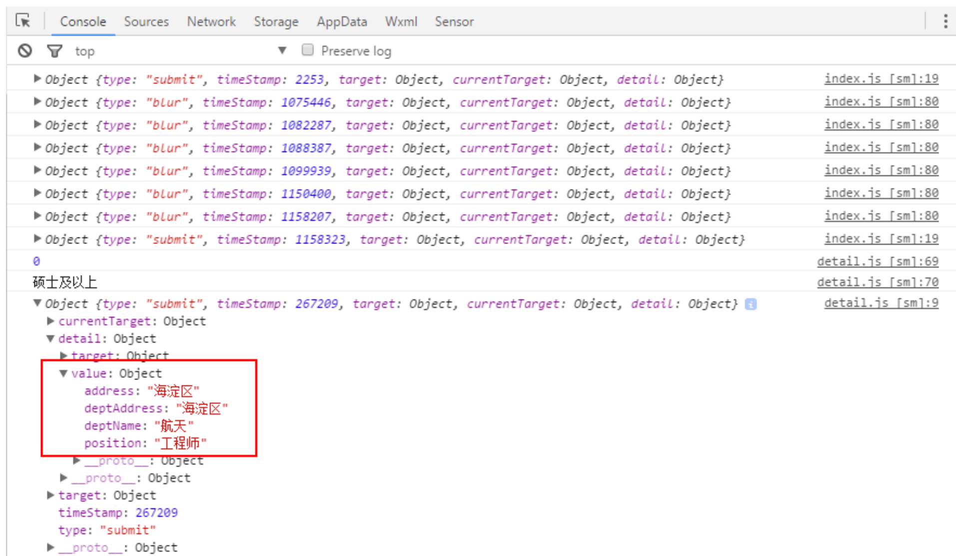


图 4.26　详细信息数据

3．联系人信息设计

（1）在 pages/contact/contact.wxml 页面文件里设计联系人信息，使用 view 视图容器组件、表单组件等进行界面布局设计，具体代码如下所示。

```
<view class="content">
<view class="tip {{info}}">{{tip}}</view>
<view class="items">
    <view class="item">
        <view><image src="/pages/images/jbxx-1.jpg" style="width:35px;height:35px;"></image>
</view>
        <view class="select">基本信息</view>
    </view>
    <view class="item dh">
        <view><image src="/pages/images/dh-1.jpg" style="width:14px;height:16px;"></image></
view>
    </view>
    <view class="item">
        <view><image src="/pages/images/xxxx-1.jpg" style="width:35px;height:35px;"></image>
</view>
        <view class="select">详细信息</view>
    </view>
    <view class="item dh">
```

```
        <view><image src="/pages/images/dh-1.jpg" style="width:14px;height:16px;"></image></
view>
    </view>
    <view class="item">
        <view><image src="/pages/images/lxrxx-1.jpg" style="width:35px;height:35px;"></image>
</view>
        <view class="select">联系人信息</view>
    </view>
</view>
<view class="menu">
亲属联系人
</view>
<view>
  <form bindsubmit="formSubmit">
  <view class="category">
    <view class="title">姓名</view>
    <view class="input"><input name="qName" placeholder="直系亲属联系人姓名" placeholder
-style="color:#999999;font-size:16px;" bindblur="checkValue"/></view>
  </view>
  <view class="line"></view>

  <view class="category">
    <view class="title">手机</view>
    <view class="input"><input name="qPhone"  placeholder="直系亲属联系人手机号码" place
holder-style="color:#999999;font-size:16px;" bindblur="checkValue"/></view>
  </view>
  <view class="line"></view>
<view class="menu">
非亲属联系人
</view>
  <view class="category">
    <view class="title">姓名</view>
    <view class="input"><input name="fqName" placeholder="非直系亲属联系人姓名" placehol
der-style="color:#999999;font-size:16px;" bindblur="checkValue"/></view>
  </view>
  <view class="line"></view>

  <view class="category">
    <view class="title">手机</view>
    <view class="input"><input name="fqPhone"  placeholder="非直系亲属联系人手机号码" pl
```

```
aceholder-style="color:#999999;font-size:16px;"  bindblur="checkValue"/></view>
   </view>

   <view  class="line"></view>
   <view  class="hr"></view>
   <button  class="btn"  form-type="submit">完成</button>
   </form>
</view>

</view>
```

（2）在 pages/contact/contact.wxss 样式文件里，给联系人信息布局添加样式，具体代码如下所示。

```
.content{
    font-family: "Microsoft YaHei";
}
.items{
   display: flex;
   flex-direction: row;
   margin: 0 auto;
   text-align: center;
   align-items: center;
   margin-top:20px;
   margin-bottom: 20px;
}
.item{
   width: 30%;
   font-size: 16px;
}
.dh{
   width: 5%;
}
.default{
   color: #999999;
}
.menu{
   height: 50px;
   background-color: #f2f2f2;
   line-height: 50px;
   padding-left: 15px;
}
```

```
.category{
   display: flex;
   flex-direction: row;
   margin-top:10px;
   margin-bottom: 10px;
   margin-left: 15px;
}
.title{
   width:30%;
}
.input{
   width:70%;
}
.line{
   border: 1px solid #cccccc;
   opacity: 0.2;
}
.btn{
   background-color: #E64340;
   color: #ffffff;
   margin-top:5px;
   width:80%;
}
.hr{
   height: 10px;
   background-color: #f2f2f2;
}
.tip{
   height: 40px;
   background-color: #E64340;
   color: #ffffff;
   font-size: 16px;
   text-align: center;
   line-height: 40px;
}
.hide{
   display: none;
}
```

（3）在 pages/contact/contact.js 业务逻辑处理文件里进行表单校验，并将数据缓存到本地，单击“完成”按钮完成表单的提交，具体代码如下所示。

```
Page({
  data: {
    info: 'hide',
    tip: ''
  },
  formSubmit: function (e) {
    console.log(e);
    var page = this;
    var card = wx.getStorageSync("card");
    var qName = e.detail.value.qName;
    if (qName == "") {
      page.setData({ info: 'show', tip: '请输入直系亲属联系人姓名！' })
      return;
    } else {
      card.qName = qName;
    }

    var qPhone = e.detail.value.qPhone;
    if (qPhone == "") {
      page.setData({ info: 'show', tip: '请输入直系亲属联系人手机号码！' })
      return;
    } else {
      card.qPhone = qPhone;
    }


    var fqName = e.detail.value.fqName;
    if (fqName == "") {
      page.setData({ info: 'show', tip: '请输入非直系亲属联系人姓名！' })
      return;
    } else {
      card.fqName = fqName;
    }

    var fqPhone = e.detail.value.fqPhone;
    if (fqPhone == "") {
      page.setData({ info: 'show', tip: '请输入非直系亲属联系人手机号码！' })
      return;
    } else {
      card.fqPhone = fqPhone;
```

```
    }

    wx.setStorageSync("card", card);
    wx.showToast({
      title: '提交成功',
      icon: 'success',
      duration: 5000,
      success:function(){
        wx.redirectTo({
          url: '../index/index',
        })
      }
    })

  },
  checkValue: function (e) {
    var name = e.detail.value;
    if (name != '') {
      this.setData({ info: 'hide' });
    }
  }

})
```

界面效果和缓存数据分别如图 4.27 和图 4.28 所示。

图 4.27　联系人信息设计

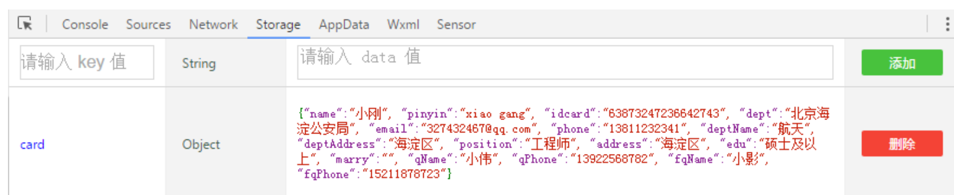

Console Sources Network Storage AppData Wxml Sensor
请输入 key 值
String
请输入 data 值
添加
card
Object
{"name":"小刚", "pinyin":"xiao gang", "idcard":"63873247236642743", "dept":"北京海淀公安局", "email":"327432467@qq.com", "phone":"13811232341", "deptName":"航天", "deptAddress":"海淀区", "position":"工程师", "address":"海淀区", "edu":"硕士及以上", "marry":"", "qName":"小伟", "qPhone":"13922568782", "fqName":"小影", "fqPhone":"15211878723"}
删除

图 4.28　本地缓存数据

第 5 章　综合案例：京东购物小程序

京东商城在小程序推出之际就发布了自己的小程序——京东购物小程序，现在已经经历了两版。第一个版本功能比较简单，仅可以领取一些优惠券和下单购买一些商品；第二个版本功能就全面多了，有商品的列表展示、商品详情页、结算页、购物车等功能。下面以京东购物小程序第二个版本为例，来设计京东购物小程序，包括首页商品列表展示设计、商品详情页设计、商品加入购物车设计、立即购买页面设计、商品检索设计、显示购物车商品设计、我的页面设计以及优惠券设计，如图 5.1、图 5.2、图 5.3 和图 5.4 所示。



图 5.1　首页



图 5.2　商品详情页



图 5.3　购物车



图 5.4　我的订单

5.1　需求描述

京东购物小程序的需求如下。

（1）底部标签导航放置 3 个菜单：首页、购物车、我的订单，菜单选中时文字颜色变为红色，图标也相应地变为红色，如图 5.5 所示。



图 5.5　底部标签导航

（2）首页搜索区域设计以及九宫格导航菜单设计，九宫格导航菜单放置 4 个菜单：领优惠券、扫码购物、物流查询和商品分类，如图 5.6 所示。



图 5.7 搜索区域及九宫格导航

（3）首页商品列表设计，通过图文列表的方式进行展现，显示商品名称、商品原价、商品现价、售出数量以及立即抢购按钮，如图 5.8 所示。



图 5.8 首页商品列表设计

（4）首页商品详情页设计，包括商品图片轮播效果、商品详细信息、“加入购物车”

和“立即抢购”按钮，如图 5.9 和图 5.10 所示。



图 5.9　商品详情一



图 5.10　商品详情二

（5）单击“加入购物车”按钮，可以将商品加入购物车里，并提示加入购物车操作成功，如图 5.11 所示。



图 5.11　加入购物车

（6）单击“立即购买”按钮，可以跳转到购买商品页面，显示商品信息以及物流配送信息，如图 5.12 和图 5.13 所示。



图 5.12 购物车中的商品信息



图 5.13 物流配送信息

（7）在首页单击“搜索框”，进入搜索页面，输入要搜索的商品，即可进行商品搜索，如图 5.14 和图 5.15 所示。



图 5.14 搜索页面



图 5.15 搜索商品

（8）在购物车页面里，显示已加入购物车里的商品，同时可以动态添加商品数量和计算商品金额，商品可以全选和单选，如图 5.16 和图 5.17 所示。



图 5.16　取消全选商品



图 5.17　全选商品

（9）在我的订单页面，通过列表式导航展示订单信息，如图 5.18 所示。



图 5.18　我的订单

（10）单击优惠券导航，可以进入优惠券页面，通过页签切换效果可以显示待使用优惠

券、已过期优惠券和已使用优惠券，如图 5.19、图 5.20 和图 5.21 所示。



图 5.19　待使用优惠券



图 5.20　已过期优惠券



图 5.21　已使用优惠券

（11）在首页，可以将首页商品列表信息分享给好友或者好友群里，如图 5.22 和图 5.23 所示。



图 5.22　分享商品列表



图 5.23　分享给好友

5.2　相关知识点

设计京东购物小程序时，由于涉及的页面和交互效果很多，所以会用到很多的微信小程序组件以及 API 接口，这些组件和 API 也是设计小程序时经常用到的内容。

1．组件相关知识点

（1）view 视图容器组件可用来进行页面布局设计，页面布局都需要使用它。

（2）swiper 滑块视图容器组件可用来进行页签切换效果设计，例如优惠券页签切换效果。

（3）icon 图标组件，用于购物车复选框设计。

（4）progress 进度条组件，用于显示首页商品售卖的数量。

（5）text 文本组件，用于文本显示。

（6）button 按钮组件，用于表单立即购买页面的提交按钮。

（7）checkbox 多项选择器组件，用于购物车复选框设计。

（8）input 输入框组件，用于商品搜索输入框。

（9）label 改进表单性组件，用于改进购物车复选框展示效果。

（10）image 图片组件，用于图片布局和显示设计。

2．API 接口相关知识点

（1）wx.request 发起网络请求后台数据，获取商品信息。

（2）wx.setStorageSync 以同步方式将数据缓存到本地，用于将商品添加到购物车，以及初始化商品列表信息，将商品列表缓存到本地。

（3）wx.getStorageSync 以同步方式获取指定 key 对应的内容，可以获取商品列表信息以及加入购物车的商品信息。

（4）wx.getLocation 获取当前的地理位置、速度，可以获取用户的位置信息。

（5）wx.showToast 显示消息提示框，当商品加入购物车成功后，显示添加成功。

（6）wx.setNavigationBarTitle 动态设置页面导航标题。

（7）wx.setNavigationBarColor 动态设置页面导航背景色。

（8）wx.navigateTo 保留当前页面，跳转到下一个页面。

（9）wx.redirectTo 关闭当前页面，跳转到应用内的某个页面。

（10）wx.navigateBack 关闭当前页面，返回上一页面或多级页面。

（11）wx.authorize 授权，部分接口需要获得用户同意后才能调用。

（12）wx.getUserInfo 获取用户信息。

（13）wx.login 登录接口。

（14）onShareAppMessage 转发分享，分享首页商品列表到好友或者好友群。

5.3 商品数据准备

京东购物小程序的商品列表数据，应该从后台服务器里获取，通过调用 wx.request 发起网络请求后台数据，获取商品信息，在这个案例中简化了这一步操作。我们在 app.js 里初始化一些商品数据，然后保存到本地缓存 goods 商品里，商品 good 有如下属性。

（1）id：商品唯一标识。

（2）pic：商品图片，用于商品列表显示时展示商品。

（3）name：商品的名称。

（4）nowPrice：商品现价。

（5）oldPrice：商品原价。

（6）soldCount：商品售出的数量。

（7）coupon：商品优惠券信息。

（8）promotion：商品促销信息。

（9）color：商品颜色或者类别。

（10）count：商品数量，默认为 1。

（11）address：商家地址。

（12）desc：商品说明。

（13）promise：商家承诺信息。

（14）photos：商品轮播图片。

（15）isjd ：是否是京东自营，1 代表自营，0 代表不是自营。

（16）size：商品尺寸。

具体代码如下所示。

```
//app.js
App({
  onLaunch: function () {
    var goods = wx.getStorageSync('goods');
    if(!goods){
        goods = this.loadGoods();
    }
    wx.setStorageSync('goods', goods);
```

```
  },
  loadGoods:function(){

    var goods = new Array();
    var good = new Object();

    good.id="1000";//商品唯一标识
    good.pic = '/images/pic/map.jpg'; //商品图片
    good.name='2017 中国地图册+世界地图册';//商品名称
    good.nowPrice='33.50';//商品现价
    good.oldPrice = '51.80';//商品原价
    good.soldCount = '31';//商品售出数量
    good.coupon = ['满 105 减 6','满 200 减 16'];//商品优惠券
    good.promotion = ['限购', '限制','满减'];//商品促销
    good.color = ['世界', '中国&世界', '世界'];//商品颜色
    good.count = 1;//商品数量
    good.address='广东省 深圳市 宝安区';//商家地址
    good.promise = ['99 元免基础运费', '京东发货&售后', '七天退货', '211 限时达', '货到付款'];
//商家承诺
    good.desc = '10:00 之前下单，预计第二天送达';//商品说明
    good.photos = ['/images/haibao/map/1.jpg', '/images/haibao/map/2.jpg', '/images/haib
ao/map/3.jpg', '/images/haibao/map/4.jpg'];//商品组图
    good.isjd = "1";//是否是京东自营，1 代表自营，0 代表不是自营
    goods[0] = good;

    var good1 = new Object();
    good1.id = "1001";//商品唯一标识
    good1.pic = '/images/pic/shoes.jpg'; //商品图片
    good1.name = '【名品秒杀 闪电发货】JOOHEES 头层牛皮定制凉鞋';//商品名称
    good1.nowPrice = '189.00';//商品现价
    good1.oldPrice = '629.00';//商品原价
    good1.soldCount = '51';//商品售出数量
    good1.coupon = ['满 199 减 40', '满 300 减 10'];//商品优惠券
    good1.promotion = ['限购', '限制', '满减'];//商品促销
    good1.color = ['白色', '黑色'];//商品颜色
    good1.size = ['35', '36', '37', '38', '39', '40'];//商品尺寸
    good1.count = 1;//商品数量
    good1.address = '广东省 深圳市 宝安区';//商家地址
    good1.promise = ['商家发货&售后', '七天退货', '货到付款', '运费险'];//商家承诺
    good1.desc = '免运费';//商品说明
```

```
    good1.photos = ['/images/haibao/shoes/1.jpg', '/images/haibao/shoes/2.jpg', '/images/
haibao/shoes/3.jpg', '/images/haibao/shoes/4.jpg', '/images/haibao/shoes/5.jpg'];//商品组图
    good1.isjd = "0";//是否是京东自营，1 代表自营，0 代表不是自营
    goods[1] = good1;

    var good2 = new Object();
    good2.id = "1002";//商品唯一标识
    good2.pic = '/images/pic/bag.jpg'; //商品图片
    good2.name = 'CAMEL 骆驼户外男女旅行 收纳防水罩背负系统登山包';//商品名称
    good2.nowPrice = '178.00';//商品现价
    good2.oldPrice = '349.00';//商品原价
    good2.soldCount = '15';//商品售出数量
    good2.coupon = ['满 299 减 20', '满 499 减 50'];//商品优惠券
    good2.promotion = ['限购', '限制'];//商品促销
    good2.color = ['黑色', '红色', '彩蓝', '墨绿'];//商品颜色
    good2.count = 1;//商品数量
    good2.address = '广东省 深圳市 宝安区';//商家地址
    good2.promise = ['商家发货&售后', '七天退货', '货到付款'];//商家承诺
    good2.desc = '店铺单笔订单不满 59 元，收运费 10 元';//商品说明
    good2.photos = ['/images/haibao/bag/1.jpg', '/images/haibao/bag/2.jpg', '/images/haib
ao/bag/3.jpg', '/images/haibao/bag/4.jpg', '/images/haibao/bag/5.jpg'];
//商品组图
    good2.isjd = "0";//是否是京东自营，1 代表自营，0 代表不是自营
    goods[2] = good2;

    var good3 = new Object();
    good3.id = "1003";//商品唯一标识
    good3.pic = '/images/pic/purifier.jpg'; //商品图片
    good3.name = 'airx A7F 净化器 甲醛 CADR 400 立方米每小时 持续吸附甲醛山包';//商品
//名称
    good3.nowPrice = '2849.00';//商品现价
    good3.oldPrice = '2899.00';//商品原价
    good3.soldCount = '23';//商品售出数量
    good3.promotion = ['加价购', '限购', '限制'];//商品促销
    good3.count = 1;//商品数量
    good3.address = '广东省 深圳市 宝安区';//商家地址
    good3.promise = ['99 元免基础运费(20kg 内)', '京东发货&售后', '七天退货', '211 限时达',
'货到付款'];//商家承诺
    good3.desc = '10：00 前下单，预计第二天送达';//商品说明
    good3.photos = ['/images/haibao/purifier/1.jpg', '/images/haibao/purifier/2.jpg', '/imag
```

```
es/haibao/purifier/3.jpg', '/images/haibao/purifier/4.jpg', '/images/haibao/purifier/5.jpg'];//商
//品组图
    good3.isjd = "1";//是否是京东自营，1 代表自营，0 代表不是自营
    goods[3] = good3;

    var good4 = new Object();
    good4.id = "1004";//商品唯一标识
    good4.pic = '/images/pic/haier.jpg'; //商品图片
    good4.name = '海尔（Haier） 1.5 匹 冷暖 变频';//商品名称
    good4.nowPrice = '3698.00';//商品现价
    good4.oldPrice = '4299.00';//商品原价
    good4.soldCount = '32';//商品售出数量
    good4.promotion = ['加价购', '限购', '限制', '优惠套装'];//商品促销
    good4.count = 1;//商品数量
    good4.color = ['大 1 匹', '1.5 匹'];//商品颜色
    good4.address = '广东省 深圳市 宝安区';//商家地址
    good4.promise = ['99 元免基础运费(20kg 内)', '京东发货&售后', '七天退货', '货到付款', '
家电净洗'];//商家承诺
    good4.desc = '18：00 前下单，预计第二天送达';//商品说明
    good4.photos = ['/images/haibao/haier/1.jpg', '/images/haibao/haier/2.jpg', '/images/h
aibao/haier/3.jpg', '/images/haibao/haier/4.jpg', '/images/haibao/haier/5.jpg'];//商品组图
    good4.isjd = "1";//是否是京东自营，1 代表自营，0 代表不是自营
    goods[4] = good4;

    return goods;

  },
  getUserInfo:function(cb){
    var that = this
    if(this.globalData.userInfo){
      typeof cb == "function" && cb(this.globalData.userInfo)
    }else{
      //调用登录接口
      wx.login({
        success: function () {
          wx.getUserInfo({
            success: function (res) {
              that.globalData.userInfo = res.userInfo
              typeof cb == "function" && cb(that.globalData.userInfo)
            }
```

```
            })
         }
      })
   }
 },
 globalData:{
   userInfo:null
 }
})
```

将商品信息保存到本地缓存里，key 值为 good，如图 5.24 所示。

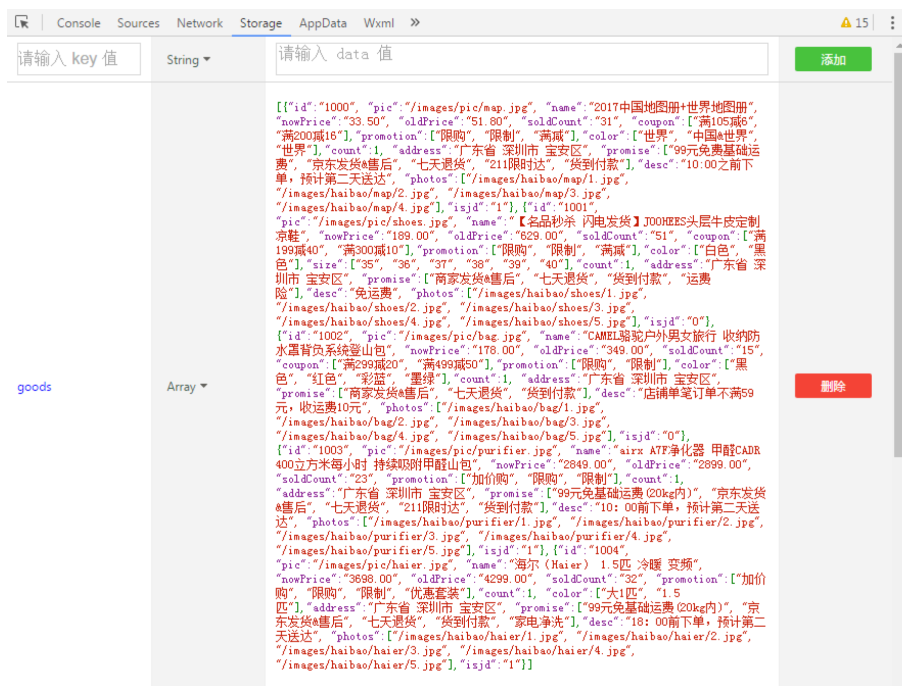


图 5.24　goods 商品列表缓存数据

5.4　首页设计

京东购物小程序首页设计包括两方面内容：一方面是商品搜索区域和九宫格导航设计，另一方面是商品列表显示区域，如图 5.25 所示。



图 5.25　首页设计

1．商品搜索区域和九宫格导航设计

（1）新建一个京东购物微信小程序，如果想在手机上预览小程序，则需要填写 AppID，AppID 可以以个人身份进行申请，把小程序需要使用的图片放置到 images 文件夹下面，在 app.js 里初始化商品数据，将其保存到本地缓存里。

（2）在 app.json 里配置底部标签导航，有三个标签导航：首页、购物车、我的订单。选中时文字颜色为红色（#E4393C），窗口背景色为红色（#C81623），导航文字为京东购物，颜色为白色，具体代码如下所示。

```
{
  "pages":[
    "pages/index/index",
    "pages/shoppingcart/shoppingcart",
    "pages/me/me"
  ],
  "window":{
    "backgroundTextStyle":"light",
    "navigationBarBackgroundColor": "#C81623",
    "navigationBarTitleText": "京东购物",
    "navigationBarTextStyle":"white"
  },
  "tabBar": {
    "backgroundColor": "#ffffff",
```

```
    "selectedColor": "#E4393C",
    "list": [{
      "pagePath": "pages/index/index",
      "text": "首页",
      "iconPath": "/images/bar/search-0.jpg",
      "selectedIconPath": "/images/bar/search-1.jpg"
    },{
      "pagePath": "pages/shoppingcart/shoppingcart",
      "text": "购物车",
      "iconPath": "/images/bar/shoppingcart-0.jpg",
      "selectedIconPath": "/images/bar/shoppingcart-1.jpg"
    },{
      "pagePath": "pages/me/me",
      "text": "我的订单",
      "iconPath": "/images/bar/order-0.jpg",
      "selectedIconPath": "/images/bar/order-1.jpg"
    }]
  }
}
```

（3）在 pages/index/index.wxml 页面文件里，设计商品搜区域以及九宫格导航菜单，具体代码如下所示。

```
<view class="content">
  <view class="bg">
    <view class="logo">
      <view class="first">JD.COM 京东</view>
      <view class="second">多 · 快 · 好 · 省</view>
    </view>
    <view class="search" bindtap="search">
      <view>
        <image src="/images/icon/search.jpg" style="width:20px;height:21px;"></image>搜索京
东商品</view>
    </view>
  </view>
<!--导航-->
<view class="nav" style="margin-top:20px;">

  <view class="item" bindtap="navBtn">
    <view>
      <image src="/images/pic/lyhq.jpg" style="width:33px;height:30px;"></image>
    </view>
```

```
    <view>
      领优惠券
    </view>
  </view>

  <view class="item" bindtap="navBtn">
    <view>
      <image src="/images/pic/smgw.jpg" style="width:33px;height:30px;"></image>
    </view>
    <view>
      扫码购物
    </view>
  </view>

  <view class="item" bindtap="navBtn">
    <view>
      <image src="/images/pic/wlcx.jpg" style="width:33px;height:30px;"></image>
    </view>
    <view>
      物流查询
    </view>
  </view>

  <view class="item" bindtap="navBtn">
    <view>
      <image src="/images/pic/spfl.jpg" style="width:33px;height:30px;"></image>
    </view>
    <view>
      商品分类
    </view>
  </view>
</view>
```

（4）在 pages/index/index.wxss 样式文件里，给商品搜区域以及九宫格导航菜单添加样式，具体代码如下所示。

```
.content{
  font-family: "Microsoft YaHei";
}
.bg{
  width: 100%;
  height: 170px;
```

```
   background-color: #C81623;
   text-align: center;
}
.logo{
   padding-top:20px;
   color: #ffffff;
}
.first{
   font-size:24px;
   font-weight: bold;
}
.second{
   font-size: 14px;
}
.search{
   height: 40px;
   width: 96%;
   background-color: #ffffff;
   margin: 0 auto;
   margin-top:30px;
   border-radius: 50px;
   line-height: 40px;
}
.search view{
   text-align: center;
   width: 100%;
   font-size: 15px;
   color: #999999;
}
.title{
    padding: 10px;
    font-size: 13px;
    color: #999999;
}
.item{
    margin-top:10px;
    text-align: center;
    font-size: 15px;
    width: 24%;
    display: inline-block;
```

```
  margin-bottom: 20px;
}
```

2．商品列表展示设计

（1）在 pages/index/index.wxml 页面文件里，设计九宫格导航下面的京东秒杀区域和商品列表显示区域，具体代码如下所示。

```
<view class="content">
  <view class="bg">
    <view class="logo">
      <view class="first">JD.COM 京东</view>
      <view class="second">多 · 快 · 好 · 省</view>
    </view>
    <view class="search" bindtap="search">
      <view>
        <image src="/images/icon/search.jpg" style="width:20px;height:21px;"></image>搜索京东商品</view>
    </view>
  </view>
<!--导航-->
<view class="nav" style="margin-top:20px;">

  <view class="item" bindtap="navBtn">
    <view>
      <image src="/images/pic/lyhq.jpg" style="width:33px;height:30px;"></image>
    </view>
    <view>
      领优惠券
    </view>
  </view>

  <view class="item" bindtap="navBtn">
    <view>
      <image src="/images/pic/smgw.jpg" style="width:33px;height:30px;"></image>
    </view>
    <view>
      扫码购物
    </view>
  </view>

  <view class="item" bindtap="navBtn">
```

```
        <view>
            <image  src="/images/pic/wlcx.jpg"  style="width:33px;height:30px;"></image>
        </view>
        <view>
            物流查询
        </view>
    </view>

    <view  class="item"  bindtap="navBtn">
        <view>
            <image  src="/images/pic/spfl.jpg"  style="width:33px;height:30px;"></image>
        </view>
        <view>
            商品分类
        </view>
    </view>
</view>
<view  class="hr"></view>
<view  class="menu">京东秒杀</view>
<!--  秒杀  -->
<view  class="seckill  ">
    <view  class="stage">
        <view  class="time">12:00</view>
        <view  style="color:#C81623">抢购中</view>
    </view>
    <view  class="stage">
        <view  class="timeDefult">14:00</view>
        <view>即将开始</view>
    </view>
    <view  class="stage">
        <view  class="timeDefult">16:00</view>
        <view>即将开始</view>
    </view>
    <view  class="stage">
        <view  class="timeDefult">18:00</view>
        <view>即将开始</view>
    </view>
    <view  class="stage">
        <view  class="timeDefult">20:00</view>
        <view>即将开始</view>
```

```
    </view>
</view>
<view class="line"></view>
<!--商品列表-->
<view class="list">
        <view class="good">
              <view class="pic">
                  <image src="/images/pic/map.jpg" mode="aspectFit" style="width:110
px;height:110px;"></image>
              </view>
              <view class="movie-info">
                  <view class="base-info">
                       <view class="name">2017 中国地图册+世界地图册</view>
                       <view class="people"><text class="price">￥26.10</text><text cl
ass="org">￥26.10</text> </view>
                        <view class="sale">
                           <view class="saleCount">已售 31%</view>
                           <view class="saleProjess"><progress percent="40" stroke-w
idth="6" color="#EC7476"/></view>
                        </view>
                        <view class="btn">立即抢购</view>
                  </view>
              </view>
         </view>
      <view class="hr"></view>
</view>
</view>
```

（2）在 pages/index/index.wxss 样式文件里，给京东秒杀区域和商品列表显示区域添加样式，具体代码如下所示。

```
.content{
  font-family: "Microsoft YaHei";
}
.bg{
  width: 100%;
  height: 170px;
  background-color: #C81623;
  text-align: center;
}
.logo{
  padding-top:20px;
```

```
    color: #ffffff;
}
.first{
    font-size:24px;
    font-weight: bold;
}
.second{
    font-size: 14px;
}
.search{
    height: 40px;
    width: 96%;
    background-color: #ffffff;
    margin: 0 auto;
    margin-top:30px;
    border-radius: 50px;
    line-height: 40px;
}
.search view{
    text-align: center;
    width: 100%;
    font-size: 15px;
    color: #999999;
}
.title{
      padding: 10px;
      font-size: 13px;
      color: #999999;
}
.item{
      margin-top:10px;
      text-align: center;
      font-size: 15px;
      width: 24%;
      display: inline-block;
      margin-bottom: 20px;
}
.hr{
    background: #E8E8ED;
    height: 10px;
```

```
}
.menu{
  text-align: center;
  margin-top:10px;
  margin-bottom: 10px;
}
.seckill{
  display: flex;
  flex-direction: row;
}
.stage{
  width: 25%;
  text-align: center;
  font-size: 12px;
}
.time{
  background-color: #C81623;
  width: 90%;
  color: #ffffff;
  height: 30px;
  line-height: 30px;
  border-radius: 20px;
  margin-left: 5px;
}
.timeDefult{
  background-color: #f2f2f2;
  width: 90%;
  height: 30px;
  line-height: 30px;
  border-radius: 20px;
  margin-left: 5px;
}
.line{
  border:1px solid #f2f2f2;
  opacity: 0.5;
  margin-top:10px;
  margin-bottom: 10px;
}
.list{
  margin-top: 10px;
```

```
  margin-bottom: 10px;
}
.good{
    display: flex;
    flex-direction: row;
    width: 90%;
    margin: 0 auto;
    border: 1px solid #cccccc;
    box-shadow: -3px 0px 9px #ececec,0px -3px 9px #ececec,0px 3px 9px #ececec,0p
x 0px 9px #ececec;
    margin-bottom: 10px;
    align-items: center;
}
.pic image{
    width:80px;
    height:100px;
    padding:10px;
}
.base-info{
    font-size: 13px;
    padding-top: 10px;
    line-height: 20px;
}
.name{
    font-size: 15px;
    font-weight: bold;
    color: #000000;
}
.people{
    color: #555555;
    margin-top: 5px;
    margin-bottom: 5px;
}
.price{
    font-size: 22px;
    color: #E53D30;
}
.org{
  text-decoration: line-through;
  margin-left: 5px;
```

```
   margin-right: 5px;
}
.desc{
      color: #333333;
}
.sale{
   display: flex;
   flex-direction: row;
   align-items: center;
}
.saleCount{
   width:40%;
   font-size: 15px;
   color: #666666;
}
.saleProjess{
   width: 60%;
}
.btn{
   background: #F46366;
   width: 100px;
   color: #ffffff;
   height: 25px;
   line-height: 25px;
   font-size: 15px;
   text-align: center;
   margin-top:10px;
   border-radius:3px;
   margin-bottom: 10px;
}
```

（3）在 pages/index/index.js 业务逻辑处理文件里，从本地缓存获取商品列表信息，定义变量 *goods* 商品列表集合，动态绑定到页面里，具体代码如下所示。

```
Page({
   data:{
      goods:[]
   },
   onLoad:function(){
      this.loadGoods();
   },
   loadGoods:function(){
```

```
      var page = this;
      var goods = wx.getStorageSync("goods");
      page.setData({goods:goods});
   }
})
```

（4）在 pages/index/index.wxml 页面里，动态渲染商品列表信息，将商品数据动态绑定到页面，具体代码如下所示。

```
<view class="content">
   <view class="bg">
      <view class="logo">
         <view class="first">JD.COM 京东</view>
         <view class="second">多  ·  快  ·  好  ·  省</view>
      </view>
      <view class="search" bindtap="search">
         <view>
            <image src="/images/icon/search.jpg" style="width:20px;height:21px;"></image>搜索京
东商品</view>
      </view>
   </view>
<!--导航-->
<view class="nav" style="margin-top:20px;">

   <view class="item" bindtap="navBtn">
      <view>
         <image src="/images/pic/lyhq.jpg" style="width:33px;height:30px;"></image>
      </view>
      <view>
         领优惠券
      </view>
   </view>

   <view class="item" bindtap="navBtn">
      <view>
         <image src="/images/pic/smgw.jpg" style="width:33px;height:30px;"></image>
      </view>
      <view>
         扫码购物
      </view>
   </view>
```

```
  <view class="item" bindtap="navBtn">
    <view>
      <image src="/images/pic/wlcx.jpg" style="width:33px;height:30px;"></image>
    </view>
    <view>
      物流查询
    </view>
  </view>

  <view class="item" bindtap="navBtn">
    <view>
      <image src="/images/pic/spfl.jpg" style="width:33px;height:30px;"></image>
    </view>
    <view>
      商品分类
    </view>
  </view>
</view>
<view class="hr"></view>
<view class="menu">京东秒杀</view>
<!-- 秒杀 -->
<view class="seckill ">
    <view class="stage">
        <view class="time">12:00</view>
        <view style="color:#C81623">抢购中</view>
    </view>
    <view class="stage">
        <view class="timeDefult">14:00</view>
        <view>即将开始</view>
    </view>
    <view class="stage">
        <view class="timeDefult">16:00</view>
        <view>即将开始</view>
    </view>
    <view class="stage">
        <view class="timeDefult">18:00</view>
        <view>即将开始</view>
    </view>
    <view class="stage">
        <view class="timeDefult">20:00</view>
```

```
        <view>即将开始</view>
    </view>
</view>
<view class="line"></view>
<!--商品列表-->
<view class="list">
 <block wx:for="{{goods}}">
<view class="good">
                  <view class="pic">
                       <image src="{{item.pic}}" mode="aspectFit" style="width:110px;heigh
t:110px;"></image>
                  </view>
                  <view class="movie-info">
                       <view class="base-info">
                             <view class="name">{{item.name}}</view>
                             <view class="people"><text class="price">￥{{item.nowPrice}}</t
ext><text class="org">￥{{item.oldPrice}}</text> </view>
                              <view class="sale">
                                  <view class="saleCount">已售 {{item.soldCount}}%</view>
                                  <view class="saleProjess"><progress percent="{{item.soldC
ount}}" stroke-width="6" color="#EC7476"/></view>
                              </view>
                              <view class="btn" id="{{item.id}}" bindtap="detail">立即抢购</vi
ew>
                       </view>
                  </view>
            </view>
</block>
</view>
</view>
```

（5）在 pages/index/index.js 业务逻辑处理文件里，使用 onShareAppMessage 转发分享函数，将商品列表页分享给好友或者好友群，具体代码如下所示。

```
Page({
  data:{
    goods:[]
  },
  onLoad:function(){
    this.loadGoods();
  },
  loadGoods:function(){
```

```
        var page = this;
        var goods = wx.getStorageSync("goods");
        page.setData({goods:goods});
    },
    onShareAppMessage: function (res) {
        return {
            title: '京东购物首页',
            path: '/page/index/index',
            success: function (res) {
                    // 转发成功
            },
            fail: function (res) {
                // 转发失败
            }
        }
    }
})
```

这样就完成了首页的设计，包括首页的搜索区域、九宫格导航区域、京东秒杀区域以及商品列表动态显示区域。

5.5　商品详情页设计

在首页商品列表里，单击立即抢购按钮，会跳转到商品详情页，显示商品的详细信息，包括商品的轮播图片、商品的名称、商品现在的价格、商品原价、商品的优惠券、商品的促销信息、商品的颜色、商品的尺寸、商品的数量、商家地址、商品描述、商家承诺信息、“加入购物车”和“立即购买”按钮，如图 5.26 和图 5.27 所示。



图 5.26　商品详情一



图 5.27　商品详情二

（1）在 app.json 里添加一个商品详情页页面路径“pages/detail/detail”，然后在 pages/index/index.js 业务逻辑处理文件里给“立即购买”按钮绑定跳转事件，单击这个按钮后会跳转到商品详情页面，具体代码如下所示。

```
Page({
  data:{
    goods:[]
  },
  onLoad:function(){
    this.loadGoods();
  },
  loadGoods:function(){
    var page = this;
    var goods = wx.getStorageSync("goods");
    page.setData({goods:goods});
  },
  onShareAppMessage: function (res) {
    return {
      title: '京东购物首页',
      path: '/page/index/index',
      success: function (res) {
```

```
            // 转发成功
        },
        fail: function (res) {
          // 转发失败
        }
      }
    },
    detail:function(e){
      console.log(e);
      var id = e.target.id;
      console.log(id);
      wx.navigateTo({
        url: '../detail/detail?id='+id,
      })
    }
})
```

（2）在 pages/detail/detail.wxml 页面文件里，进行商品详情页的布局设计，具体代码如下所示。

```
<view class="content">
  <view class="haibao">
    <swiper indicator-dots="{{indicatorDots}}" autoplay="{{autoplay}}" interval="{{interval}}" dura
tion="{{duration}}" indicator-color="#f2f2f2" indicator-active-color="#C81623" style="height:350px;
">
      <block wx:for="{{imgUrls}}">
        <swiper-item>
          <image src="{{item}}" class="silde-image"></image>
        </swiper-item>
      </block>
    </swiper>
  </view>

  <view class="btn">
    <view class="nowPrice">￥33.50</view>
    <view class="oldPrice">
      <view class="old">￥51.80</view>
      <view class="seckill">京东秒杀</view>
    </view>
  </view>
  <view class="title">【自营】2017 中国地图册+世界地图册</view>
  <view class="hr"></view>
```

```
<view class="promotion">
  <view>领券</view>
  <view class="item">满 105 减 6</view>
  <view class="item">满 200 减 16</view>
</view>
<view class="hr"></view>
<view class="promotion">
  <view>促销</view>
  <view class="item">限购</view>
  <view class="item">限制</view>
</view>
<view class="hr"></view>
<view class="desc">已选中国&世界</view>
<view class="line"></view>
<view class="condition">
  <text class="conditionName">颜色</text>
  <view class="tips">
    <view class="tip select">世界</view>
    <view class="tip">中国&世界</view>
    <view class="tip">中国</view>
    <view class="tip">世界</view>
    <view class="tip">中国&世界</view>
    <view class="tip">中国</view>
  </view>
</view>

<view class="condition">
  <text class="conditionName">尺寸</text>
  <view class="tips">
    <view class="tip select">尺寸 1</view>
    <view class="tip">尺寸 2</view>
    <view class="tip">尺寸 3</view>
    <view class="tip">尺寸 4</view>
    <view class="tip">尺寸 5</view>
    <view class="tip">尺寸 6</view>
  </view>
</view>

<view class="condition">
  <text class="conditionName">数量</text>
```

```
    <view class="tips">
      <view class="priceInfo">
        <view class="minus">-</view>
        <view class="count">1</view>
        <view class="add">+</view>
      </view>
    </view>
  </view>
<view class="hr"></view>
<view class="desc">地址 广东省 深圳市 宝安区</view>
<view class="line"></view>
<view class="desc"><text class="xh">[现货] </text> <text class="xhdesc">10:00 之前下单，第二天可送到</text></view>
<view class="line"></view>
<view class="promise">
<icon type="success" size="15" color="red"/>99 元免费基础运费
<icon type="success" size="15" color="red"/>京东发货&售后
<icon type="success" size="15" color="red"/>七天退货
<icon type="success" size="15" color="red"/>211 限时达
<icon type="success" size="15" color="red"/>货到付款
</view>
<view class="hr"></view>
<view style="height:50px;"></view>
<view class="buyBtns">
  <view class="shoppingcart">
   <view class="num">1</view>
   <image src="/images/icon/cart_no_num.jpg" style="width:45px;height:39px;"></image>
  </view>
  <view class="intocart">加入购物车</view>
  <view class="buy">立即购买</view>
</view>
</view>
```

（3）在 pages/detail/detail.wxss 样式文件里，给商品详情页的布局添加样式，具体代码如下所示。

```
.content{
  font-family: "Microsoft YaHei";
}
.haibao{
    text-align: center;
    width: 100%;
```

```
}
.silde-image{
    width: 100%;
    height: 350px;
}
.btn{
  background-color: #ED4D57;
  height: 40px;
  display: flex;
  flex-direction: row;
}
.nowPrice{
  width:70%;
  line-height: 40px;
  font-size: 28px;
  color: #ffffff;
  text-align: center;
}
.oldPrice{
  width:30%;
  font-size: 13px;
  color: #ffffff;
}
.old{
  text-decoration: line-through;
  font-size: 15px;
}
.seckill{
  border: 1px solid #ffffff;
  width:60px;
  text-align: center;
}
.title{
  margin:10px;
}
.hr{
  height: 10px;
  background-color: #f2f2f2;
}
.promotion{
```

```
  display: flex;
  flex-direction: row;
  margin:10px;
  align-items: center;
  font-size: 16px;
}
.item{
  border: 1px solid #ED4D57;
  color: #ED4D57;
  font-size: 13px;
  height: 20px;
  line-height: 20px;
  padding-left:5px;
  padding-right: 5px;
  margin-left: 5px;
  border-radius: 3px;
}
.desc{
  margin: 10px;
  font-size: 16px;
}
.line{
  border:1px solid #f2f2f2;
  opacity: 0.5;
}
.condition{
  margin: 10px;
  font-size: 16px;
  color: #999999;
  clear: both;
}
.conditionName{
  margin-right: 20px;
  position: absolute;
}
.tips{
  margin-left: 40px;
}
.tip{
    background-color: #E8E8ED;
```

```
        height:25px;
        line-height: 25px;
        border-radius: 3px;
        text-align: center;
        font-size: 15px;
        margin-right: 10px;
        float: left;
        margin-bottom: 10px;
        padding-left: 10px;
        padding-right: 10px;
}
.select{
    background-color: #ED4D57;
    color: #ffffff;
}
.priceInfo{
        display: flex;
        flex-direction: row;
        margin-top:10px;
        height: 30px;
}
.minus,.add{
        border: 1px solid #cccccc;
        width: 25px;
        text-align: center;
        line-height: 30px;
}
.count{
        border-top: 1px solid #cccccc;
        border-bottom: 1px solid #cccccc;
        width: 40px;
        text-align: center;
        line-height: 30px;
}
.xh{
    color: #ED4D57;
    font-size: 15px;
}
.xhdesc{
    color: #999999;
```

```
  font-size: 15px;
}
.promise{
  font-size: 14px;
  margin:10px;
}
.promise icon{
  margin-left:10px;
}
.buyBtns{
  position: fixed;
  bottom: 0px;
  height: 40px;
  width: 100%;
  display: flex;
  flex-direction: row;
}
.shoppingcart{
  width:16%;
  background-color: #ffffff;
  text-align: center;
}
.intocart{
  width:42%;
  background-color: #FF9600;
  color: #ffffff;
  text-align: center;
  line-height: 40px;

}
.buy{
  width:42%;
  background-color: #E3393C;
  color: #ffffff;
  text-align: center;
  line-height: 40px;
}
.num{
  z-index: 999;
  width:16px;
```

```
  height: 16px;
  border-radius: 50%;
  background-color: #E3393C;
  position: absolute;
  left: 35px;
  bottom:27px;
  font-size: 12px;
  color: #ffffff;
}
```

（4）在 pages/detail/detail.js 业务逻辑处理文件里，初始化海报轮播的数据以及商品详情的数据，具体代码如下所示。

```
Page({
  data: {
    indicatorDots: true,
    autoplay: true,
    interval: 5000,
    duration: 1000,
    imgUrls: [],
    good:{},
    quantity:0
  },
  onLoad:function(e){
    var id = e.id;
    this.initData(id);
    var orders = wx.getStorageSync("orders");
    this.setData({ quantity: orders.length});
  },
  initData:function(id){//渲染商品信息
    var goods = wx.getStorageSync("goods");
    var good = {};
    for (var i = 0; i < goods.length; i++) {
      var g = goods[i];
      if (g.id == id) {
        var isjd = g.isjd;
        if(isjd=="1"){
          g.name = "【自营】" + g.name;
        }
        good = g;
        break;
      }
```

```
    }
    console.log(good);
    var page = this;
    page.setData({ imgUrls: good.photos,good:good});
  }
})
```

（5）在 pages/detail/detail.wxml 页面文件里，将初始化海报轮播的数据以及商品详情的数据动态绑定到页面，具体代码如下所示。

```
<view class="content">
  <view class="haibao">
    <swiper indicator-dots="{{indicatorDots}}" autoplay="{{autoplay}}" interval="{{interval}}" dura
tion="{{duration}}" indicator-color="#f2f2f2" indicator-active-color="#C81623" style="height:350px;
">
      <block wx:for="{{imgUrls}}">
        <swiper-item>
          <image src="{{item}}" class="silde-image"></image>
        </swiper-item>
      </block>
    </swiper>
  </view>
  <view class="btn">
    <view class="nowPrice">￥{{good.nowPrice}}</view>
    <view class="oldPrice">
      <view class="old">￥{{good.oldPrice}}</view>
      <view class="seckill">京东秒杀</view>
    </view>
  </view>
  <view class="title">{{good.name}}</view>
  <view class="hr"></view>
  <block wx:if="{{good.coupon.length > 0}}">
    <view class="promotion">
      <view>领券</view>
      <block wx:for="{{good.coupon}}">
        <view class="item">{{item}}</view>
      </block>
    </view>
    <view class="hr"></view>
  </block>

  <block wx:if="{{good.promotion.length > 0}}">
```

```
    <view class="promotion">
      <view>促销</view>
      <block wx:for="{{good.promotion}}">
        <view class="item">{{item}}</view>
      </block>
    </view>
    <view class="hr"></view>
  </block>

  <view class="desc">已选
    <block wx:if="{{good.color.length > 0}}">
      {{good.color[0]}}
    </block>
    <block wx:if="{{good.size.length > 0}}">
      {{good.size[0]}}
    </block>
    ，1 个
  </view>
  <view class="line"></view>

  <block wx:if="{{good.color.length > 0}}">
    <view class="condition">
      <text class="conditionName">颜色</text>
      <view class="tips">
        <block wx:for="{{good.color}}">
          <view wx:if="{{index ==0}}" class="tip select"> {{item}} </view>
          <view wx:else class="tip"> {{item}} </view>
        </block>
      </view>
    </view>
  </block>
  <block wx:if="{{good.size.length > 0}}">
    <view class="condition">
      <text class="conditionName">尺寸</text>
      <view class="tips">
        <block wx:for="{{good.size}}">
          <view wx:if="{{index ==0}}" class="tip select"> {{item}} </view>
          <view wx:else class="tip"> {{item}} </view>
        </block>
      </view>
```

```
    </view>
  </block>
  <view class="condition">
    <text class="conditionName">数量</text>
    <view class="tips">
      <view class="priceInfo">
        <view class="minus">-</view>
        <view class="count">{{good.count}}</view>
        <view class="add">+</view>
      </view>
    </view>
  </view>
  <view class="hr"></view>
  <view class="desc">地址 {{good.address}}</view>
  <view class="line"></view>
  <view class="desc"><text class="xh">[现货] </text> <text class="xhdesc">{{good.des
c}}</text></view>
  <view class="line"></view>
  <view class="promise">
    <block wx:for="{{good.promise}}">
      <icon type="success" size="15" color="red" />{{item}}
    </block>
  </view>
  <view class="hr"></view>
  <view style="height:50px;"></view>
  <view class="buyBtns">
    <view class="shoppingcart">
      <view class="num">{{quantity}}</view>
      <image src="/images/icon/cart_no_num.jpg" style="width:45px;height:39px;"></im
age>
    </view>
    <view class="intocart" bindtap="addGood" id="{{good.id}}">加入购物车</view>
    <view class="buy" id="{{good.id}}" bindtap="buy">立即购买</view>
  </view>
</view>
```

（6）在 pages/detail/detail.js 业务逻辑处理文件里，给“加入购物车”按钮和“立即购买”按钮添加绑定事件，在 app.json 里添加立即购买页面路径“pages/buy/buy”，然后单击“立即购买”按钮后跳转到购买商品页面，具体代码如下所示。

```
Page({
  data: {
```

```
    indicatorDots: true,
    autoplay: true,
    interval: 5000,
    duration: 1000,
    imgUrls: [],
    good:{},
    quantity:0
  },
  onLoad:function(e){
    var id = e.id;
    this.initData(id);
    var orders = wx.getStorageSync("orders");
    this.setData({ quantity: orders.length});
  },
  initData:function(id){//渲染商品信息
    var goods = wx.getStorageSync("goods");
    var good = {};
    for (var i = 0; i < goods.length; i++) {
      var g = goods[i];
      if (g.id == id) {
        var isjd = g.isjd;
        if(isjd=="1"){
          g.name = "【自营】" + g.name;
        }
        good = g;
        break;
      }
    }
    console.log(good);
    var page = this;
    page.setData({ imgUrls: good.photos,good:good});
  },
  addGood:function(e){//添加商品到购物车
    console.log(e);
    var id = e.target.id;
    var orders = wx.getStorageSync("orders");
    var flag=true;
    var newOrders=[];
    if (orders){//先判断购物车里是否有该商品，如果有，就在数量上加 1
       for(var i=0;i<orders.length;i++){
```

```
            var order = orders[i];
            if(id==order.id){
                order.quantity = order.quantity +1;
                flag=false;
            }
            newOrders.push(order);
        }
    }
    if(flag){//如果购物车里没有该商品，就添加到购物车里
      var goods = wx.getStorageSync("goods");
      for(var i=0;i<goods.length;i++){
            var good = goods[i];
            if(id==good.id){
              good.quantity=1;
              newOrders.push(good);
              break;
            }
      }
    }
    wx.setStorageSync("orders", newOrders);//将商品保存到本地数据
    wx.showToast({//提示保存成功
      title: '成功',
      icon: 'success',
      duration: 2000
    });
    var page = this;
    page.setData({ quantity: newOrders.length});//购物车数量显示
  },
  buy:function(e){//直接购买
    var id = e.target.id;
    wx.navigateTo({
      url: '../buy/buy?id='+id,
    })
  }
})
```

这样就完成了商品的详情页设计，先设计商品详情页的布局设计，然后添加样式，根据首页传递过来的商品 id 值从缓存数据里获取数据，动态地绑定到页面；加入购物车操作是将该商品保存到本地缓存的 orders 里，添加商品成功后，有相应的提示。

5.6　立即购买页面

在商品详情页面里，单击“立即购买”按钮后，会跳转到立即购买页面。在立即购买页面里会显示收货地址、商品名称、商品数量、配送方式、商品金额、运费等信息，如图 5.28 和图 5.29 所示。



图 5.28　立即购买一



图 5.29　立即购买二

（1）在 pages/buy/buy.wxml 页面文件里，可对立即购买商品页面进行设计，具体代码如下所示。

```
<view class="content">
  <view class="hr"></view>
  <view class="addressInfo" bindtap="selectAddress">
    <view class="contactInfo">
      刘凯文 13811267546
    </view>
    <view class="address">
      <view class="desc">[默认]
        <text class="addressDetail">北京市丰台区四环到五环之间丰台区郭公庄地铁口郭公庄中
街 30 号院</text>
```

```
        </view>
        <view class="detail">></view>
      </view>
    </view>
    <view class="hr"></view>
    <view class="info">
      <view class="menu">
        JD 京东自营
      </view>
      <view class="line"></view>
      <view class="good">
        <view class="pic">
          <image src="/images/pic/map.jpg" mode="aspectFit" style="width:110px;height:110px;
"></image>
        </view>
        <view class="movie-info">
          <view class="base-info">
            <view class="name">2017 中国地图册+世界地图册</view>
            <view class="quantity">数量：1 件</view>
            <view class="people">
              <view class="price">￥26.10</view>
              <view class="priceInfo">
                <view class="minus">-</view>
                <view class="count">1</view>
                <view class="add">+</view>
              </view>
            </view>
            <view class="sale">
              <view class="saleCount">支持 7 天无理由退货</view>
              <view class="saleCount">价格说明</view>
            </view>
          </view>
        </view>
      </view>
      <view class="line"></view>
      <view class="item">
        <view class="menu">配送方式</view>
        <view class="menuInfo">配送方式</view>
      </view>
      <view class="item">
```

```
        <view class="menu">中小件</view>
        <view class="menuInfo">送货时间：第二天送达</view>
    </view>
    <view class="hr"></view>
    <view class="item">
        <view class="menu">发票信息</view>
        <view class="menuInfo">个人（纸质发票） 不开发票 ></view>
    </view>
    <view class="line"></view>
    <view class="item">
        <view class="menu">京东券</view>
        <view class="menuInfo">无可用 ></view>
    </view>
    <view class="hr"></view>
    <view class="item">
        <view class="menu">商品金额</view>
        <view class="amountInfo">￥178.00</view>
    </view>
    <view class="item">
        <view class="menu">运费</view>
        <view class="amountInfo">+￥0.00</view>
    </view>
    <view class="item">
        <view class="menu">优惠券</view>
        <view class="amountInfo">-￥0.00</view>
    </view>
    <view class="line"></view>
    <view class="result">
        <text>实付金额：</text>
        <text class="amount">￥178.00</text>
    </view>

    <button type="primary" style="width:95%; margin-top:10px;">微信支付</button>
    <button type="primary" style="width:95%; background:#f2f2f2;margin-top:10px; color: #000
000;">货到付款</button>
  </view>
</view>
```

（2）在 pages/buy/buy.wxss 样式文件里，给立即购买商品页面添加样式，具体代码如下所示。

```
.content{
```

```
    font-family: "Microsoft YaHei";
    height: 900px;
    background-color: #F9F9F8;
}
.hr{
    height: 12px;
    background-color: #F9F9F8;
}
.addressInfo{
  height: 100px;
  background-color: #ffffff;
}
.address{
    display: flex;
    flex-direction: row;
    width: 90%;
}
.contactInfo{
  padding: 10px;
}
.desc{
   line-height: 25px;
   padding-left: 10px;
   font-size: 15px;
   color: #ff0000;
}
.detail{
    position: absolute;
    right: 10px;
    line-height: 25px;
}
.addressDetail{
  color: #000000;
}
.info{
   background-color: #ffffff;
}
.menu{
  padding:10px;
  width:40%;
```

```
  font-size: 18px;
}
.line{
    border: 1px solid #cccccc;
    opacity: 0.2;
}
.good{
    display: flex;
    flex-direction: row;
    width: 90%;
    margin: 0 auto;
    margin-bottom: 10px;
    align-items: center;
}
.pic image{
    width:80px;
    height:100px;
    padding:10px;
}
.base-info{
    font-size: 13px;
    padding-top: 10px;
    line-height: 20px;
}
.name{
    font-size: 15px;
    font-weight: bold;
    color: #000000;
}
.quantity{
  margin-top:10px;
  margin-bottom: 10px;
  font-size: 15px;
  color: #999999;
}
.people{
    color: #555555;
    margin-bottom: 5px;
    display: flex;
    flex-direction: row;
```

```
        height: 50px;
}
.price{
        font-size: 22px;
        color: #E53D30;
        line-height: 50px;
        margin-right: 10px;
}
.sale{
    display: flex;
    flex-direction: row;
    align-items: center;
}
.saleCount{
    font-size: 13px;
    color: #666666;
    margin-right: 10px;
}
.priceInfo{
        display: flex;
        flex-direction: row;
        margin-top:10px;
        height: 30px;
}
.minus,.add{
        border: 1px solid #cccccc;
        width: 25px;
        text-align: center;
        line-height: 30px;
}
.count{
        border-top: 1px solid #cccccc;
        border-bottom: 1px solid #cccccc;
        width: 40px;
        text-align: center;
        line-height: 30px;
}
.item{
    display: flex;
    flex-direction: row;
```

```
}
.menuInfo{
   width: 60%;
   text-align: right;
   line-height: 40px;
   font-size: 15px;
   color: #999999;
   margin-right: 10px;
}
.amountInfo{
   width: 60%;
   text-align: right;
   line-height: 40px;
   font-size: 15px;
   color: #ED4D57;
   font-weight: bold;
   margin-right: 10px;
}
.result{
   text-align: right;
   margin: 10px;
   width: 95%;
   font-size: 18px;
}
.amount{
   font-size: 22px;
   color: #ED4D57;
   font-weight: bold;
}
.btn{
   width:95%;
   margin-top:10px;
}
.btn2{
   width:95%;
   background:#f2f2f2;
   margin-top:10px;
   color: #000000;
}
```

（3）在 pages/buy/buy.js 业务逻辑处理文件里，根据商品的 id 到本地缓存数据里查找商

品的详细数据，具体代码如下所示。

```
Page({
  data:{
    good:{}
  },
  onLoad: function (e) {
    console.log(e);
    var id = e.id;
    console.log(id);
    var goods = wx.getStorageSync("goods");
    for (var i = 0; i < goods.length; i++) {
      var good = goods[i];
      if (id == good.id) {
        this.setData({ good: good });
        break;
      }
    }
  }
})
```

（4）在 pages/buy/buy.wxml 页面文件里，将商品 good 信息动态绑定到页面，具体代码如下所示。

```
<view class="content">
  <view class="hr"></view>
  <view class="addressInfo" bindtap="selectAddress">
    <view class="contactInfo">
      刘凯文 13811267546
    </view>
    <view class="address">
      <view class="desc">[默认]
        <text class="addressDetail">北京市丰台区四环到五环之间丰台区郭公庄地铁口郭公庄中街 30 号院</text>
      </view>
      <view class="detail">></view>
    </view>
  </view>

  <view class="hr"></view>
  <view class="info">
    <view class="menu">
      JD 京东自营
```

```
    </view>
    <view class="line"></view>
    <view class="good">
      <view class="pic">
        <image src="/images/pic/map.jpg" mode="aspectFit" style="width:110px;height:
110px;"></image>
      </view>
      <view class="movie-info">
        <view class="base-info">
          <view class="name">{{good.name}}</view>
          <view class="quantity">数量：{{good.count}}件</view>
          <view class="people">
            <view class="price">￥{{good.nowPrice}}</view>
            <view class="priceInfo">
              <view class="minus">-</view>
              <view class="count">{{good.count}}</view>
              <view class="add">+</view>
            </view>
          </view>
          <view class="sale">
            <view class="saleCount">支持 7 天无理由退货</view>
            <view class="saleCount">价格说明</view>
          </view>
        </view>
      </view>
    </view>
    <view class="line"></view>
    <view class="item">
      <view class="menu">配送方式</view>
      <view class="menuInfo">配送方式</view>
    </view>
    <view class="item">
      <view class="menu">中小件</view>
      <view class="menuInfo">送货时间：第二天送达</view>
    </view>
    <view class="hr"></view>
    <view class="item">
      <view class="menu">发票信息</view>
      <view class="menuInfo">个人（纸质发票） 不开发票 ></view>
    </view>
```

```
    <view class="line"></view>
    <view class="item">
      <view class="menu">京东券</view>
      <view class="menuInfo">无可用 ></view>
    </view>
    <view class="hr"></view>
    <view class="item">
      <view class="menu">商品金额</view>
      <view class="amountInfo">￥{{good.nowPrice}}</view>
    </view>
    <view class="item">
      <view class="menu">运费</view>
      <view class="amountInfo">+￥0.00</view>
    </view>
    <view class="item">
      <view class="menu">优惠券</view>
      <view class="amountInfo">-￥0.00</view>
    </view>
    <view class="line"></view>
    <view class="result">
      <text>实付金额：</text>
      <text class="amount">￥{{good.nowPrice}}</text>
    </view>

    <button type="primary" style="width:95%; margin-top:10px;">微信支付</button>
    <button type="primary" style="width:95%; background:#f2f2f2;margin-top:10px; col
or: #000000;">货到付款</button>
  </view>
</view>
```

这样就完成了立即购买页面的设计，先设计立即购买的静态页面布局及样式，然后根据商品 id 值从本地缓存数据中获取商品信息，再动态绑定到页面里。

5.7　商品搜索设计

京东购物小程序的首页里，有搜索商品的区域，单击搜索商品的输入框，会跳转到搜索商品页面，在这个页面里可以输入要搜索的商品并展示一些入门的搜索标签，搜索出的商品会以列表展现的方式进行展现，如图 5.30 和图 5.31 所示。

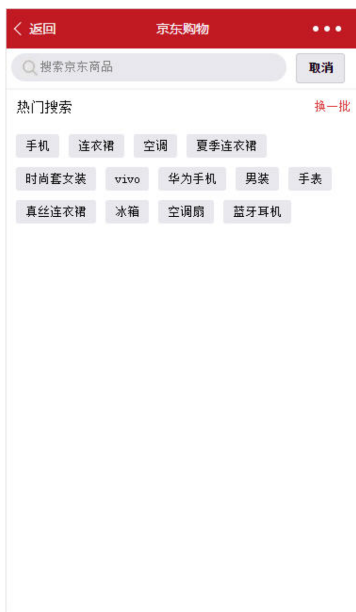


图 5.30　搜索页面



图 5.31　搜索商品

（1）在 app.json 里添加搜索页面路径“pages/search/search”，在 pages/index/index.js 业务逻辑处理文件里，当单击首页搜索区域时，会跳转到搜索页面，具体代码如下所示。

```
Page({
  data:{
    goods:[]
  },
  onLoad:function(){
    this.loadGoods();
  },
  loadGoods:function(){
    var page = this;
    var goods = wx.getStorageSync("goods");
    page.setData({goods:goods});
  },
  detail:function(e){
    console.log(e);
    var id = e.target.id;
    console.log(id);
    wx.navigateTo({
      url: '../detail/detail?id='+id,
```

```
    })
  },
  onShareAppMessage: function (res) {
    return {
      title: '京东购物首页',
      path: '/page/index/index',
      success: function (res) {
          // 转发成功
      },
      fail: function (res) {
        // 转发失败
      }
    }
  },
  search:function(){
    wx.navigateTo({
      url: '../search/search'
    })
  }
})
```

（2）在 pages/search/search.wxml 页面文件里，进行搜索商品的输入框以及入门搜索商品标签设计，具体代码如下所示。

```
<view class="search">
  <view class="searchBg">
    <view>
      <image src="/images/icon/search-1.jpg" style="width:20px;height:21px;"></image>
    </view>
    <view>
      <input type="text" placeholder="搜索京东商品" placeholder-class="holder" bindblur="sear
chGoods" id="search" value="{{name}}"/>
    </view>
  </view>
  <view class="btn" bindtap="resetSearch">取消</view>
</view>
<view class="hr"></view>
<view class="hotSearch">
    <view class="title">
      <view class="left">热门搜索</view>
      <view class="right">换一批</view>
    </view>
```

```
    <view class="tips">
      <view class="tip">手机</view>
      <view class="tip">连衣裙</view>
      <view class="tip">空调</view>
      <view class="tip">夏季连衣裙</view>
      <view class="tip">时尚套女装</view>
      <view class="tip">vivo</view>
      <view class="tip">华为手机</view>
      <view class="tip">男装</view>
      <view class="tip">手表</view>
      <view class="tip">真丝连衣裙</view>
      <view class="tip">冰箱</view>
      <view class="tip">空调扇</view>
      <view class="tip">蓝牙耳机</view>
    </view>
</view>
```

（3）在 pages/search/search.wxss 样式文件里，给搜索商品的输入框以及入门搜索商品标签设计添加样式，具体代码如下所示。

```
.search{
    display: flex;
    flex-direction: row;
    padding:5px;
}
.searchBg{
    background-color: #E8E8ED;
    width:80%;
    border-radius:15px;
    height: 30px;
    display: flex;
    flex-direction: row;
}
.searchBg image{
    margin-left: 10px;
    margin-top: 5px;
}
.search input{
    height: 30px;
    line-height: 30px;
}
.holder{
```

```
    font-size: 13px;
}
.btn{
    font-size: 13px;
    font-weight: bold;
    line-height: 30px;
    margin-left: 10px;
    border: 1px solid #cccccc;
    width: 50px;
    text-align: center;
    background-color: #E8E8ED;
    border-radius:3px;
}
.hr{
    border: 1px solid #cccccc;
    opacity: 0.2;
}
.title{
    display: flex;
    flex-direction: row;
    padding: 10px;
}
.left{
    width: 80%;
    font-size: 15px;
}
.right{
    width: 20%;
    font-size: 13px;
    color: #E4393C;
    text-align: right;
}
.tips{
    padding:10px;
}
.tip{
    background-color: #E8E8ED;
    height:25px;
    line-height: 25px;
    border-radius: 3px;
```

```
        text-align: center;
        font-size: 13px;
        margin-right: 10px;
        float: left;
        margin-bottom: 10px;
        padding-left: 10px;
        padding-right: 10px;
}
```

（4）在 pages/search/search.wxml 页面文件里，输入要搜索的商品，这样满足条件的商品就会以列表的形式展现出来，具体代码如下所示。

```
<view class="search">
    <view class="searchBg">
        <view>
            <image src="/images/icon/search-1.jpg" style="width:20px;height:21px;"></image>
        </view>
        <view>
            <input type="text" placeholder="搜索京东商品" placeholder-class="holder" bindblur="sear
chGoods" id="search" value="{{name}}"/>
        </view>
    </view>
    <view class="btn" bindtap="resetSearch">取消</view>
</view>
<view class="hr"></view>
<view class="hotSearch">
        <view class="title">
            <view class="left">热门搜索</view>
            <view class="right">换一批</view>
        </view>
        <view class="tips">
            <view class="tip">手机</view>
            <view class="tip">连衣裙</view>
            <view class="tip">空调</view>
            <view class="tip">夏季连衣裙</view>
            <view class="tip">时尚套女装</view>
            <view class="tip">vivo</view>
            <view class="tip">华为手机</view>
            <view class="tip">男装</view>
            <view class="tip">手表</view>
            <view class="tip">真丝连衣裙</view>
            <view class="tip">冰箱</view>
```

```
        <view  class="tip">空调扇</view>
        <view  class="tip">蓝牙耳机</view>
    </view>
</view>
<view  class="item">
  <view  class="name">空调机</view>
  <view  class="hr"></view>
</view>
<view  class="item">
  <view  class="name">空调扇</view>
  <view  class="hr"></view>
</view>
```

（5）在 pages/search/search.wxss 样式文件里，给搜索商品结果添加样式，具体代码如下所示。

```
.search{
    display:  flex;
    flex-direction:  row;
    padding:5px;
}
.searchBg{
    background-color:  #E8E8ED;
    width:80%;
    border-radius:15px;
    height:  30px;
    display:  flex;
    flex-direction:  row;
}
.searchBg  image{
    margin-left:  10px;
    margin-top:  5px;
}
.search  input{
    height:  30px;
    line-height:  30px;
}
.holder{
    font-size:  13px;
}
.btn{
    font-size:  13px;
```

```
    font-weight: bold;
    line-height: 30px;
    margin-left: 10px;
    border: 1px solid #cccccc;
    width: 50px;
    text-align: center;
    background-color: #E8E8ED;
    border-radius:3px;
}
.hr{
    border: 1px solid #cccccc;
    opacity: 0.2;
}
.title{
    display: flex;
    flex-direction: row;
    padding: 10px;
}
.left{
    width: 80%;
    font-size: 15px;
}
.right{
    width: 20%;
    font-size: 13px;
    color: #E4393C;
    text-align: right;
}
.tips{
    padding:10px;
}
.tip{
    background-color: #E8E8ED;
    height:25px;
    line-height: 25px;
    border-radius: 3px;
    text-align: center;
    font-size: 13px;
    margin-right: 10px;
    float: left;
```

```
    margin-bottom: 10px;
    padding-left: 10px;
    padding-right: 10px;
}
.item{
    width: 100%;
    padding-left:10px;
    padding-right:10px;
    font-size: 15px;
    padding-top: 10px;
}
.name{
    margin-bottom: 10px;
}
```

（6）在 pages/search/search.js 业务逻辑处理文件里，添加要搜索的商品和返回首页操作，具体代码如下所示。

```
Page({
  data: {
    result: [],
    name:''
  },
  searchGoods: function (e) {//搜索商品
    var name = e.detail.value;
    var goods = this.loadGoods();
    var result = new Array();
    if (name != '') {
      for (var i = 0; i < goods.length; i++) {
        var good = goods[i];
        if (good.indexOf(name) > -1) {
          result.push(good);
        }
      }
    }
    console.log(result);
    this.setData({ result: result });
  },
  loadGoods: function () {
    var goods = ['八宝粥京东自营', '八宝粥原料', '八宝粥米', '八宝粥箱', '八宝粥杂粮', '八宝粥银鹭', '八宝粥料', '空调扇', '空调挂机', '空调被', '空调柜机', '空调京东自营', '空调扇 制冷', '空调挡风被',];
```

```
        return goods;
    },
    resetSearch:function(){//返回上一级页面
        wx.navigateBack({
            delta: 1
        })
    }

})
```

（7）在 pages/search/search.wxml 页面文件里，默认显示搜索输入框和搜索热门标签，如果输入商品搜索，那么就显示搜索商品结果列表，具体代码如下所示。

```
<view class="search">
    <view class="searchBg">
        <view>
            <image src="/images/icon/search-1.jpg" style="width:20px;height:21px;"></image>
        </view>
        <view>
            <input type="text" placeholder="搜索京东商品" placeholder-class="holder" bindblur="sear
chGoods" id="search" value="{{name}}" />
        </view>
    </view>
    <view class="btn" bindtap="resetSearch">取消</view>
</view>
<view class="hr"></view>
<block wx:if="{{result.length > 0}}">
    <block wx:for="{{result}}">
        <view class="item">
            <view class="name">{{item}}</view>
            <view class="hr"></view>
        </view>
    </block>
</block>
<block wx:else>
    <view class="hotSearch">
        <view class="title">
            <view class="left">热门搜索</view>
            <view class="right">换一批</view>
        </view>
        <view class="tips">
            <view class="tip">手机</view>
```

```
        <view  class="tip">连衣裙</view>
        <view  class="tip">空调</view>
        <view  class="tip">夏季连衣裙</view>
        <view  class="tip">时尚套女装</view>
        <view  class="tip">vivo</view>
        <view  class="tip">华为手机</view>
        <view  class="tip">男装</view>
        <view  class="tip">手表</view>
        <view  class="tip">真丝连衣裙</view>
        <view  class="tip">冰箱</view>
        <view  class="tip">空调扇</view>
        <view  class="tip">蓝牙耳机</view>
      </view>
    </view>
</block>
```

商品搜索设计时要注意，默认显示的是搜索的输入框和热门搜索标签，当输入商品进行搜索时就会显示搜索商品结果列表，通过条件判断来控制显示哪些内容。

5.8　显示购物车商品设计

商品加入到购物车后，在购物车页面里会显示商品信息，包括商品的图片、商品的名称、商品的数量、商品的价格等，可以动态地添加或者减少商品的数量，也可以勾选商品或者全选商品，如图 5.32 所示。



图 5.32　购物车

（1）在 app.json 里添加购物车页面路径“pages/shoppingcart/shoppingcart”，在 pages/index/index.wxml 页面文件里，进行购物车商品列表的布局设计，具体代码如下所示。

```
<view class="content">
  <view class="info">
    <view class="line"></view>
    <view class="receive">
      京东购物
    </view>
    <view class="line"></view>
    <view class="items">
      <checkbox-group bindchange="checkboxChange">


        <view class="item">
          <view class="icon">
            <checkbox checked/>
          </view>
          <view class="pic">
            <image src="/images/order/1.jpg" style="width:80px;height:80px;"></image>
          </view>
```

```
      <view class="order">
        <view class="title">【京东超市】伊利中老年奶粉听装 900g(新老包装)</view>
        <view class="desc">
          <view>重量：1.200kg</view>
          <view>规格：</view>
        </view>
        <view class="priceInfo">
          <view class="price">￥86.00</view>
          <view class="minus">-</view>
          <view class="count">1</view>
          <view class="add">+</view>
        </view>
      </view>
    </view>
    <view class="line"></view>
    <view class="item">
      <view class="icon">
        <checkbox checked/>
      </view>
      <view class="pic">
        <image src="/images/order/2.jpg" style="width:80px;height:80px;"></image>
      </view>
      <view class="order">
        <view class="title">TP-Link TP-WN726N 免驱版 外置天线 USB 无线网卡</view>
        <view class="desc">
          <view>重量：0.100kg</view>
          <view>规格：外置天线</view>
        </view>
        <view class="priceInfo">
          <view class="price">￥49.00</view>
          <view class="minus">-</view>
          <view class="count">1</view>
          <view class="add">+</view>
        </view>
      </view>
    </view>
    <view class="line"></view>
    <view class="item">
      <view class="icon">
        <checkbox checked/>
```

```
          </view>
          <view class="pic">
            <image src="/images/order/3.jpg" style="width:80px;height:80px;"></image>
          </view>
          <view class="order">
            <view class="title">Apple iPhone7（A1660）128GB 金色移动联通电信 4G 手机</vi
ew>
            <view class="desc">
              <view>重量：0.390kg/件</view>
              <view>规格：金色,128GB</view>
            </view>
            <view class="priceInfo">
              <view class="price">￥5499.00</view>
              <view class="minus">-</view>
              <view class="count">1</view>
              <view class="add">+</view>
            </view>
          </view>
        </view>
        <view class="line"></view>
      </checkbox-group>
      <checkbox-group bindchange="checkAll">
        <view class="totalInfo">
          <view class="all">
            <view>
                <label for="boxAll">
                <checkbox checked="{{selectedAll}}" id="boxAll" hidden/>
                <icon type="{{selectedAll==true?'success':'circle'}}" color="#E4393C" data-valu
e="{{item.id}}" size="20"/>
             </label>

            </view>
            <view>
              全选
            </view>
          </view>
          <view class="amount">
            <view class="total">
              总计：￥{{totalPrice}}元
            </view>
```

```
                <view>
                    不含运费，已优惠￥0.00
                </view>
            </view>
            <view class="opr">去结算</view>
        </view>
    </checkbox-group>
    </view>
  </view>
</view>
```

（2）在 pages/index/index.wxss 样式文件里，给购物车商品列表的布局设计添加样式，具体代码如下所示。

```
.content{
    font-family: "Microsoft YaHei";
    height: 600px;
    background-color: #F9F9F8;
}
.info{
   background-color: #ffffff;
}
.line{
    border: 1px solid #cccccc;
    opacity: 0.2;
}
.receive{
    display: flex;
    flex-direction: row;
    padding: 10px;
}
.item{
    display: flex;
    flex-direction: row;
    padding: 10px;
    align-items: center;
}
.order{
    width:100%;
    height: 87px;
}
.title{
```

```
    font-size: 15px;
}
.desc{
    display: flex;
    flex-direction: row;
    font-size: 13px;
    color: #cccccc;
}
.desc view{
    margin-right: 10px;
}
.priceInfo{
    display: flex;
    flex-direction: row;
    margin-top:10px;
}
.price{
    width: 65%;
    font-size: 15px;
    color: #ff0000;
    text-align: left;
}
.minus,.add{
    border: 1px solid #cccccc;
    width: 25px;
    text-align: center;
}
.count{
    border-top: 1px solid #cccccc;
    border-bottom: 1px solid #cccccc;
    width: 40px;
    text-align: center;
}
.totalInfo{
    display: flex;
    flex-direction: row;
    height: 60px;
}
.all{
    align-items: center;
```

```
        padding-left: 10px;
        width: 20%;
        font-size: 12px;
        margin-top: 10px;
}
.amount{
        width:50%;
        font-size: 13px;
        text-align: right;
}
.total{
        font-size: 16px;
        color: #ff0000;
        font-weight: bold;
        margin-bottom: 10px;
}
.opr{
        position: absolute;
        right: 0px;
        width: 92px;
        font-size: 15px;
        font-weight: bold;
        background-color: #E4393C;
        height: 60px;
        text-align: center;
        line-height: 60px;
        color:#ffffff;
}
.icon{
    margin-right: 10px;
}
```

（3）在 pages/index/index.js 业务逻辑处理文件里，初始化购物车里的商品，添加加载购物车商品的函数、商品复选框函数、全选复选框函数、添加商品数量函数和减少商品数量函数，具体代码如下所示。

```
Page({
    data:{
        orders:[],//加入购物车里的商品集合
        selectedAll:false,//全选按钮标志位，true 代表全选选中，false 代表全选未选中
        totalPrice:0  //总金额
    },
```

```
  onLoad:function(options){
    this.loadOrders();
    wx.setNavigationBarTitle({//动态修改页面标题文字
      title: '购物车'
    })
    wx.setNavigationBarColor({
      frontColor: '#000000',//导航文字颜色
      backgroundColor: '#ffffff',//导航背景色
      animation: {//动画效果
        duration: 400,
        timingFunc: 'easeIn'
      }
    })
  },
  loadOrders:function(){ //加载购物车里的商品
    var orders = wx.getStorageSync('orders');//从本地缓存数据 orders 里获取数据
     var newOrders = [];
     var totalPrice=0;
     var selectedAll = true;
      for(var i=0;i < orders.length;i++){
         var order = orders[i];
         if (order.selected){//购物车里的每件商品都有一个 selected 属性，selected 等于 true 时，
//代表这件商品被选中，要计算金额
           totalPrice += order.nowPrice * order.quantity;//计算选中商品的金额
         }else{
           selectedAll = false;//购物车里的商品，如果有一件是未选中的，则 selectedAll 全选标
//志位就等于 false
         }
         newOrders.push(order);
      }
      wx.setStorageSync("orders", newOrders);//重新加入缓存
      this.setData({ totalPrice: totalPrice, orders: newOrders, selectedAll: selectedAll});//数据
//绑定到页面里
  },
  checkboxChange:function(e){//每件商品前的复选框操作函数
    var ids = e.detail.value;//会把选中的复选框的 id 值以数组集合的形式传递过来
    var orders = wx.getStorageSync('orders');
    var totalPrice=0;
    var newOrders = [];
    for(var i=0;i < orders.length;i++){
```

```
        var order = orders[i];
        var flag = true;
         for(var j=0;j < ids.length;j++){
             if(order.id == ids[j]){//传递过来的 ids 数组集合值，都是选中的商品，需要计算总的金
//额
               totalPrice += order.nowPrice * order.quantity;
               order.selected = true;//代表该件商品是选中状态
               flag = false;//代表该件商品是选中状态
             }
         }
         if (flag) {//代表该件商品是未选中状态
           order.selected = false;
         }
         newOrders.push(order);
      }
      wx.setStorageSync("orders", newOrders);//重新加入缓存数据
      this.loadOrders();//重新加载页面
   },
   checkAll:function(e){//全选复选框操作函数
      var orders = wx.getStorageSync("orders");
      console.log(e);
      var newOrders = [];
      var selectedAll = this.data.selectedAll;
      for(var i=0;i<orders.length;i++){
          var order = orders[i];
          if (selectedAll){//如果当前状态值是全选中，那么再单击的时候，全选复选框应该为未选中
//状态
            order.selected = false;
          }else{
            order.selected = true;
          }
          newOrders.push(order);
      }
      wx.setStorageSync("orders", newOrders)//重新加入缓存数据
      this.loadOrders();//重新加载页面
   },
   addOrders: function (e) {//添加商品数量函数
      var id = e.currentTarget.id;
      var orders = wx.getStorageSync('orders');
      var addOrders = new Array();
```

```
    for (var i = 0; i < orders.length; i++) {
      var order = orders[i];
      if (order.id == id) {
        var quantity = order.quantity;
        order.quantity = quantity + 1;//找到该件商品数量加 1
      }
      addOrders[i] = order;
    }

    wx.setStorageSync('orders', addOrders);//重新加入缓存数据
    this.loadOrders();//重新加载页面
  },
  minusOrders: function (e) {//减少商品数量函数
    console.log(e);
    var id = e.currentTarget.id;

    var orders = wx.getStorageSync('orders');
    var addOrders = new Array();
    var add = true;
    for (var i = 0; i < orders.length; i++) {
      var order = orders[i];
      if (order.id == id) {
        var count = order.quantity;
        if(count >= 2){
          order.quantity = count - 1;//找到该件商品数量减 1
        }
      }
      addOrders[i] = order;
    }
    wx.setStorageSync('orders', addOrders);//重新加入缓存数据
    this.loadOrders();//重新加载页面
  }
})
```

（4）在 pages/index/index.wxml 页面文件里，动态渲染购物车里的商品，具体代码如下所示。

```
<view class="content">
  <view class="info">
    <view class="line"></view>
    <view class="receive">
      京东购物
```

```
    </view>
    <view class="line"></view>
    <view class="items">
      <checkbox-group bindchange="checkboxChange">
        <block wx:for="{{orders}}">
          <view class="item">
            <view class="icon">
            <label for="{{item.id}}">
               <checkbox id="{{item.id}}" value="{{item.id}}"  checked="{{item.selecte
d}}" hidden/>
               <icon type="{{item.selected==true?'success':'circle'}}" color="#E4393C"
data-value="{{item.id}}" size="20"/>
           </label>

            </view>
            <view class="pic">
              <image src="{{item.pic}}" style="width:80px;height:80px;"></image>
            </view>
            <view class="order">
              <view class="title">{{item.name}}</view>
              <view class="desc">
                <view>数量：{{item.quantity}}</view>
              </view>
              <view class="priceInfo">
                <view class="price">￥{{item.nowPrice}}</view>
                <view class="minus" id="{{item.id}}" bindtap="minusOrders">-</view>
                <view class="count">{{item.quantity}}</view>
                <view class="add" id="{{item.id}}" bindtap="addOrders">+</view>
              </view>
            </view>
          </view>
          <view class="line"></view>
        </block>
      </checkbox-group>
      <checkbox-group bindchange="checkAll">
        <view class="totalInfo">
          <view class="all">
            <view>
               <label for="boxAll">
               <checkbox checked="{{selectedAll}}" id="boxAll" hidden/>
```

```
                <icon type="{{selectedAll==true?'success':'circle'}}" color="#E4393C" dat
a-value="{{item.id}}" size="20"/>
            </label>

            </view>
            <view>
              全选
            </view>
          </view>
          <view class="amount">
            <view class="total">
              总计：￥{{totalPrice}}元
            </view>
            <view>
              不含运费，已优惠￥0.00
            </view>
          </view>
          <view class="opr">去结算</view>
        </view>
      </checkbox-group>
    </view>
  </view>
</view>
```

重点要掌握 label 改进表单性组件的使用，可以使复选框变为圆形，通过复选框可以动态计算商品的金额，以及添加商品数量、减少商品数量业务逻辑的处理。

5.9 我的订单页面设计

京东购物小程序的我的订单页面设计是通过列表式导航的方式来设计导航菜单的，导航菜单有全部订单、待付款、待收货、售后订单和优惠券 5 个导航菜单，如图 5.33 所示。



图 5.33　我的订单

（1）在 pages/me/me.wxml 页面文件里，进行列表式导航设计，具体代码如下所示。

```
<view  class="content">
  <view  class="bg">
    <view  class="head">
      <view  class="headIcon">
        <image  src="/images/icon/head.jpg"  style="width:99px;height:99px;"></image>
      </view>
      <view  class="login">
        <view>{{userInfo.nickName}}</view>
        <view  class="account">账号：356725727_k</view>
      </view>
    </view>
  </view>
  <view  class="hr"></view>
  <view  class="item">
    <view  class="img">
      <image  src="/images/icon/qbdd.jpg"  style="width:20px;height:24px;"></image>
    </view>
    <view  class="name">全部订单</view>
    <view  class="detail">
```

```
      <text>></text>
    </view>
  </view>
  <view class="line"></view>
  <view class="item">
    <view class="img">
      <image src="/images/icon/dfk.jpg" style="width:20px;height:24px;"></image>
    </view>
    <view class="name">待付款</view>
    <view class="detail">
      <text>></text>
    </view>
  </view>
  <view class="line"></view>
  <view class="item">
    <view class="img">
      <image src="/images/icon/dsh.jpg" style="width:23px;height:20px;"></image>
    </view>
    <view class="name">待收货</view>
    <view class="detail">
      <text class="count">2 单</text>
      <text> ></text>
    </view>
  </view>
  <view class="line"></view>
  <view class="item">
    <view class="img">
      <image src="/images/icon/shdd.jpg" style="width:25px;height:24px;"></image>
    </view>
    <view class="name">售后订单</view>
    <view class="detail">
      <text> ></text>
    </view>
  </view>
  <view class="hr"></view>
  <view class="item" bindtap="seeCoupon">
    <view class="img">
      <image src="/images/icon/yhq.jpg" style="width:25px;height:22px;"></image>
    </view>
    <view class="name">优惠券</view>
```

```
    <view  class="detail">
      <text  class="count">4 张</text>
      <text>  ></text>
    </view>
  </view>
  <view  class="line"></view>
</view>
```

（2）在 pages/me/me.wxss 样式文件里，给列表式导航设计添加样式，具体代码如下所示。

```
.content{
    background-color:  #E8E8ED;
    height:  600px;
}
.bg{
  width:100%;
  height:  150px;
  background-color:  #F54844;
}
.head{
    display:  flex;
    flex-direction:  row;
}
.headIcon{
    margin:  10px;
}
.login{
    color:  #ffffff;
    font-size:  15px;
    font-weight:  bold;
    position:  absolute;
    left:120px;
    margin-top:30px;
}
.account{
    color:  #f2f2f2;
    font-size:  13px;
    margin-top:10px;
}
.item{
    display:flex;
```

```
        flex-direction:row;
        background-color: #ffffff;
}
.hr{
        width: 100%;
        height: 15px;
}
.img{
        margin-left:10px;
        line-height: 50px;
}
.name{
        padding-top:15px;
        padding-left: 10px;
        padding-bottom:15px;
        font-size:15px;
}
.detail{
        font-size: 15px;
        position: absolute;
        right: 10px;
        height: 50px;
        line-height: 50px;
        color: #888888;
}
.line{
        border: 1px solid #cccccc;
        opacity: 0.2;
}
.count{
        font-size: 13px;
        color: #ff0000;
}
```

（2）在 pages/me/me.wxjs 业务逻辑处理文件里，用户授权后可获取用户信息，同时修改页面导航文字和背景色，具体代码如下所示。

```
Page({
    data:{
        userInfo:{}
    },
    onLoad:function(){
```

```
    var page = this;
    wx.getSetting({// 可以通过 wx.getSetting 先查询一下用户是否授权了 "scope.record" 这个
//scope
      success(res) {
        if (!res['scope.userInfo']) {
          wx.authorize({//获取用户授权
            scope: 'scope.userInfo',
            success() {
              // 用户同意小程序获取用户信息后，后续调用 wx.getUserInfo 接口就不会弹出窗
//询问
                wx.getUserInfo({
                  success: function (res) {
                    var userInfo = res.userInfo
                    page.setData({userInfo:userInfo});
                  }
              })
            }
          })
        }
      }
    })
    wx.setNavigationBarTitle({//动态修改页面标题文字
      title: '我的'
    })
    wx.setNavigationBarColor({
      frontColor: '#000000',//导航文字颜色
      backgroundColor: '#ffffff',//导航背景色
      animation: {//动画效果
        duration: 400,
        timingFunc: 'easeIn'
      }
    })
  }
})
```

列表式导航是经常会用到的一种导航方式，通过这种导航方式，可以把多个导航在页面里按顺序显示出来，用户需要使用哪个导航，直接单击进入即可。

5.10　优惠券设计

在我的订单页面里有优惠券列表导航，单击优惠券导航会进入优惠券二级页面，显示优惠券信息，包括待使用优惠券、已过期优惠券和已使用优惠券，通过页签切换效果动态展示这些优惠券信息，如图 5.34、图 5.35 和图 5.36 所示。



图 5.34　待使用优惠券



图 5.35　已过期优惠券



图 5.36　已使用优惠券

（1）在 app.json 里添加优惠券页面路径“pages/coupon/coupon”，在 pages/me/me.js 里添加跳转到优惠券页面的函数，具体代码如下所示。

```
Page({
  data:{
    userInfo:{}
  },
  onLoad:function(){
    var page = this;
    wx.getSetting({// 可以通过 wx.getSetting 先查询一下用户是否授权了 "scope.record" 这个
//scope
      success(res) {
        if (!res['scope.userInfo']) {
```

```
                wx.authorize({//获取用户授权
                  scope: 'scope.userInfo',
                  success() {
                    // 用户同意小程序获取用户信息后，后续调用 wx.getUserInfo 接口时不会弹窗询问
                      wx.getUserInfo({
                        success: function (res) {
                          var userInfo = res.userInfo
                          page.setData({userInfo:userInfo});
                        }
                    })
                  }
                })
              }
            }
          })
          wx.setNavigationBarTitle({//动态修改页面标题文字
            title: '我的'
          })
          wx.setNavigationBarColor({
            frontColor: '#000000',//导航文字颜色
            backgroundColor: '#ffffff',//导航背景色
            animation: {//动画效果
              duration: 400,
              timingFunc: 'easeIn'
            }
          })
        },
        seeCoupon: function () {
          wx.navigateTo({
            url: '../coupon/coupon'
          })
        }
})
```

（2）在 pages/me/me.wxml 页面文件里，进行优惠券界面布局设计以及页签切换效果设计，具体代码如下所示。

```
<view class="content">
  <view class="menu">
    <view class="{{currentTab==0?'select':'default'}}" data-current="0" bindtap="switchNav">待使用 4</view>
    <view class="{{currentTab==1?'select':'default'}}" data-current="1" bindtap="switchNav">已
```

```
过期 3</view>
    <view class="{{currentTab==2?'select':'default'}}" data-current="2" bindtap="switchNav">已
使用 1</view>
  </view>
  <view class="hr"></view>
  <block wx:for="{{coupons}}">
    <block wx:if="{{currentTab==0}}">
      <view class="item">
        <view class="priceInfo">
          <view class="price">
            <text class="rmb">￥</text>
            <text class="count">{{item.price}}</text>
          </view>
          <view class="type">
            <view>
              <image src="/images/icon/dongquan.jpg" style="width:40px;height:16px;"></ima
ge> {{item.condition}}
            </view>
            <view>{{item.goods}}</view>
          </view>
        </view>
        <view class="desc">
          <view class="left">{{item.way}}</view>
          <view class="right">{{item.date}}</view>
        </view>
      </view>
    </block>
    <block wx:else>
      <view class="item1">
        <view class="priceInfo">
          <view class="price1">
            <text class="rmb">￥</text>
            <text class="count">{{item.price}}</text>
          </view>
          <view class="type1">
            <view>
              <image src="/images/icon/dongquan-1.jpg" style="width:40px;height:16px;"></im
age> {{item.condition}}
            </view>
            <view>{{item.goods}}</view>
```

```
          </view>
        </view>
        <view class="desc1">
          <view class="left">{{item.way}}</view>
          <view class="right">{{item.date}}</view>
        </view>
      </view>
    </block>

  </block>
</view>
```

（2）在 pages/coupon/coupon.wxml 页面文件里，进行优惠券界面布局设计以及页签切换效果设计，具体代码如下所示。

```
<view class="content">
  <view class="menu">
    <view class="{{currentTab==0?'select':'default'}}" data-current="0" bindtap="switchNav">待
使用 4</view>
    <view class="{{currentTab==1?'select':'default'}}" data-current="1" bindtap="switchNav">已
过期 3</view>
    <view class="{{currentTab==2?'select':'default'}}" data-current="2" bindtap="switchNav">已
使用 1</view>
  </view>
  <view class="hr"></view>
  <block wx:for="{{coupons}}">
    <block wx:if="{{currentTab==0}}">
      <view class="item">
        <view class="priceInfo">
          <view class="price">
            <text class="rmb">￥</text>
            <text class="count">{{item.price}}</text>
          </view>
          <view class="type">
            <view>
              <image src="/images/icon/dongquan.jpg" style="width:40px;height:16px;"></ima
ge> {{item.condition}}
            </view>
            <view>{{item.goods}}</view>
          </view>
        </view>
        <view class="desc">
```

```
                <view class="left">{{item.way}}</view>
                <view class="right">{{item.date}}</view>
            </view>
        </view>
    </block>
    <block wx:else>
        <view class="item1">
            <view class="priceInfo">
                <view class="price1">
                    <text class="rmb">￥</text>
                    <text class="count">{{item.price}}</text>
                </view>
                <view class="type1">
                    <view>
                        <image src="/images/icon/dongquan-1.jpg" style="width:40px;height:16px;"></im
age> {{item.condition}}
                    </view>
                    <view>{{item.goods}}</view>
                </view>
            </view>
            <view class="desc1">
                <view class="left">{{item.way}}</view>
                <view class="right">{{item.date}}</view>
            </view>
        </view>
    </block>

  </block>
</view>
```

（3）在 pages/coupon/coupon.wxss 样式文件里，给优惠券界面布局设计以及页签切换效果设计添加样式，具体代码如下所示。

```
.content{
   font-family: "Microsoft YaHei";
}
.menu{
    display: flex;
    flex-direction: row;
    width: 100%;
    border-bottom: 1px solid #f2f2f2;
}
```

```
.menu view{
      margin: 0 auto;
}
.select{
      font-size:15px;
      color: red;
      width: 33%;
      text-align: center;
      height: 45px;
      line-height: 45px;
      border-bottom:5rpx solid red;
}
.default{
      width: 33%;
      font-size:15px;
      text-align: center;
      height: 45px;
      line-height: 45px;
}

.item{
   border: 1px solid #47C1C4;
   width: 96%;
   height: 100px;
   margin: 0 auto;
   border-radius:5px;
   margin-top:10px;
}
.item1{
   border: 1px solid #CCCCCC;
   width: 96%;
   height: 100px;
   margin: 0 auto;
   border-radius:5px;
   margin-top:10px;
}
.priceInfo{
   display: flex;
   flex-direction: row;
   padding:5px;
```

```
    height:  60px;
}
.price{
    width:  20%;
    line-height:  60px;
    color:  #47C1C4;
}
.price1{
    width:  20%;
    line-height:  60px;
    color:  #CCCCCC;
}
.type{
    width:80%;
    font-size:  15px;
    color:  #47C1C4;
    display:  flex;
    flex-direction:  column;
}
.type1{
    width:80%;
    font-size:  15px;
    color:  #CCCCCC;
    display:  flex;
    flex-direction:  column;
}
.type  view{
        width:  100%;
        margin-top:5px;
}
.rmb{
    font-size:  13px;
}
.count{
    font-size:  20px;
}
.desc{
    font-size:  12px;
    background-color:  #47C1C4;
    height:  30px;
```

```
   color: #ffffff;
   line-height: 30px;
   display: flex;
   flex-direction: row;
}
.desc1{
   font-size: 12px;
   background-color: #CCCCCC;
   height: 30px;
   color: #ffffff;
   line-height: 30px;
   display: flex;
   flex-direction: row;
}
.left{
      width:60%;
      text-align: left;
      margin-left: 10px;
}
.right{
      width:40%;
      text-align: right;
      margin-right: 10px;
}
```

（4）在 pages/coupon/coupon.js 业务逻辑处理文件里，初始化优惠券信息以及页签信息，具体代码如下所示。

```
Page({
   data: {
      currentTab: 0,
      coupons: []
   },
   onLoad: function (options) {
      var coupons = this.loadCoupons(0);
      this.setData({coupons:coupons});
   },
   switchNav: function (e) {
      var page = this;
      var index = e.target.dataset.current;
      if (this.data.currentTab == index) {
         return false;
```

```
  } else {
    page.setData({ currentTab: index });
  }
  var coupons = this.loadCoupons(index);
  page.setData({coupons:coupons});
},
loadCoupons: function (flag) {
  var coupons = new Array();
  var coupon = new Object();
  coupon.id = "1";
  coupon.price = "200";
  coupon.condition = "满 1000 可用";
  coupon.goods = "仅可购买网络品类指定商品";
  coupon.way = "全平台";
  coupon.date = "2017.3.22-2017.12.22";
  coupon.type="0";
  coupons.push(coupon);

  var coupon2 = new Object();
  coupon2.id = "2";
  coupon2.price = "100";
  coupon2.condition = "满 500 可用";
  coupon2.goods = "仅可购买网络品类指定商品";
  coupon2.way = "全平台";
  coupon2.date = "2017.3.22-2017.12.22";
  coupon2.type="0";
  coupons.push(coupon2);

  var coupon3 = new Object();
  coupon3.id = "3";
  coupon3.price = "50";
  coupon3.condition = "满 100 可用";
  coupon3.goods = "仅可购买网络品类指定商品";
  coupon3.way = "全平台";
  coupon3.date = "2017.3.22-2017.12.22";
  coupon3.type="0";
  coupons.push(coupon3);

  var coupon4 = new Object();
  coupon4.id = "4";
```

```
coupon4.price = "200";
coupon4.condition = "满 800 可用";
coupon4.goods = "仅可购买国内机票商品";
coupon4.way = "全平台";
coupon4.date = "2017.3.22-2017.12.22";
coupon4.type="0";
coupons.push(coupon4);

var coupon5 = new Object();
coupon5.id = "5";
coupon5.price = "300";
coupon5.condition = "满 5199 可用";
coupon5.goods = "仅可购买自营 iPhone 手机部分商品";
coupon5.way = "全平台";
coupon5.date = "2017.1.16-2017.1.26";
coupon5.type="1";
coupons.push(coupon5);

var coupon6 = new Object();
coupon6.id = "6";
coupon6.price = "30";
coupon6.condition = "满 500 可用";
coupon6.goods = "全品类(特例商品除外)";
coupon6.way = "全平台";
coupon6.date = "2017.1.1-2017.1.3";
coupon6.type="1";
coupons.push(coupon6);

var coupon7 = new Object();
coupon7.id = "7";
coupon7.price = "10";
coupon7.condition = "满 200 可用";
coupon7.goods = "全品类(特例商品除外)";
coupon7.way = "全平台";
coupon7.date = "2016.12.22-2017.12.25";
coupon7.type="1";
coupons.push(coupon7);

var coupon8 = new Object();
coupon8.id = "8";
```

```
        coupon8.price = "50";
        coupon8.condition = "满 598 可用";
        coupon8.goods = "限购[劳士顿官方旗舰店]店铺商品";
        coupon8.way = "全平台";
        coupon8.date = "2017.1.3-2017.1.31";
        coupon8.type="2";
        coupons.push(coupon8);
        var result = new Array();
        for(var i=0;i<coupons.length;i++){
            if(flag==coupons[i].type){
                result.push(coupons[i]);
            }
        }
        return result;
    }

})
```

这样就完成了优惠券设计，通过页签切换效果，可以在待使用优惠券、已过期优惠券和已使用优惠券间直接进行切换展示。

5.11　小结

京东购物小程序是一个相对完整的案例，它完成了京东购物的主要功能和业务，用到了很多微信小程序的组件和微信小程序的 API。通过京东购物页面的布局设计，帮助读者学会利用这些组件来进行页面布局并熟悉这些组件的使用场景；通过页面样式设计，来学会如何构建自己想要的页面样式，利用微信小程序的 API 来完成各种业务逻辑处理以及效果处理。读者可结合组件、样式和 API 的使用，构建出自己的小程序。